



David Filipe Serra Henriques

Licenciatura em Ciências de Engenharia Eletrotécnica e de Computadores

Sensores “em movimento”

Dissertação para obtenção do Grau de Mestre em
Engenharia Eletrotécnica e de Computadores

Orientador: Tiago Oliveira Cardoso, Doutor, FCT



FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

[Setembro 2015]

Sensores “em movimento”

Copyright © David Filipe Serra Henriques (32229), Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa

A Faculdade de Ciências e Tecnologia e a Universidade Nova de Lisboa têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objetivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

Agradecimentos

Agradeço ao meu Professor Orientador Tiago por me ter dado motivação para continuar e entregar tudo a tempo devido.

Agradeço ao Sr. Eng. Rui Henriques, por ajudar na evolução deste projeto, tanto nas ideias como concretização desta dissertação.

À minha namorada agradeço por me ter ajudar a completar esta fase da minha vida, por me ter ajudado a ultrapassar os obstáculos e pela paciência que teve ao longo de todos estes anos.

Aos meus pais por me terem dado a oportunidade de poder chegar até aqui, por me ajudarem, darem tudo sempre que necessitava e estarem sempre presentes em todos os momentos importantes da minha vida.

Agradeço a meu grande amigo Diogo Joshua por me ter acompanhado e apoiado sempre que precisei ao fim destes anos todos e a dar-me inspiração para continuar.

Por fim e não menos importante, agradeço também ao meu grande amigo, Ricardo Lima, que desde o secundário que estamos juntos nesta grande batalha (sempre com piadas e grande ajuda), inclusive no projeto completo desta dissertação.

Resumo

A domótica e a Internet das Coisas (*Internet of Things* – IoT) são dois conceitos que estão a ter bastante impacto na sociedade e a mudar o seu quotidiano, pelo que o utilizador é mais exigente na utilização dos seus equipamentos, pretendendo, que sejam mais simples e eficazes.

A IoT é também muito utilizada na vertente da qualidade de vida (verificação da qualidade do ar, temperatura etc.) pelo que é necessário a utilização de uma rede de sensores de forma que toda a informação recolhida seja disponibilizada ao utilizador. Os sensores estão ligados a nós da rede de forma a recolherem informação. Essa rede pode ser constituída por diferentes topologias, sendo que a mais usual é a rede em Malha, pois no caso de um dos nós deixar de funcionar, a mensagem é enviada através de outros nós até chegar ao nó principal. O passo seguinte consiste em enviar toda essa informação para a *Cloud*, sendo que, nas soluções existentes, é necessário que o utilizador tenha um ponto fixo (*gateway*) com acesso à internet. Quando não é possível internet por cabo, a solução é a utilização de redes sem fios ou a utilização de, por exemplo, um cartão 3G/4G que implica o pagamento de taxas às operadoras móveis. Estas soluções implicam, na sua grande maioria uma instalação elétrica para alimentação dos nós dos sensores.

O grande problema surge quando o utilizador não possui nenhum acesso à Internet (no local onde são instalados os sensores), ou no caso de não existência de nenhuma instalação elétrica para alimentação dos nós dos sensores.

A solução proposta nesta dissertação consiste na utilização de um telemóvel, de um utilizador aleatório, como *gateway*. Assim, um utilizador comum pode ligar os seus sensores onde for necessário. Esses sensores são configurados *Over-The-Air* através de uma camada de aplicação do

dispositivo de comunicação, como por exemplo, o Synapse. Após configuração, os sensores estão prontos a recolher toda a informação e enviá-la através da rede até ao nó principal. O nó principal é constituído pelo dispositivo de comunicação referido anteriormente, ligado a um microcontrolador (Arduino, por exemplo) que tem agregado um leitor de cartões SD e um dispositivo Bluetooth (BLE). Toda a informação recolhida pelos sensores é guardada no cartão SD até que um utilizador, com um telemóvel (Smartphone Android com a aplicação desenvolvida instalada) com o Bluetooth ligado, se aproxime do nó principal. Assim que a ligação é aceite e estabelecida, a aplicação envia a data mais recente de um sensor específico presente na sua base de dados na *Cloud* para o Arduino, permitindo que este apague os dados mais antigos presentes no cartão e envie, como resposta para o telemóvel, a informação mais antiga recolhida pelo sensor, atualizando assim a informação.

A aplicação deste conceito pode ser útil quando não existe nenhuma ligação à internet ou quando um utilizador, por exemplo uma entidade responsável pelo meio ambiente e que seja necessário inserir sensores numa floresta, para prevenção de fogos. Assim os nós vão enviar toda a informação recolhida através da sua rede. Posteriormente, cabe ao utilizador escolher um sítio estratégico, onde saiba que irão passar indivíduos com alguma frequência, de modo a que estes recebam essa informação para o seu telemóvel.

Palavras-chave: Redes de Sensores sem Fios, Synapse, ZigBee, IEEE 802.15.4, Bluetooth, BLE, Arduino, Domótica, Internet das Coisas, Gateway, Plataformas Móveis, *Cloud*

Abstract

Domotics and the *Internet of Things* (IoT) are two concepts which are having much impact on our society and changing our quotidian, making the user much more exigent about their devices, that means, the devices have to be easier and more efficient for the user.

IOT it's also used on the life quality (as quality air check, temperature, etc.) and a wireless sensor network is needed to collect all information passing by the sensors and finally to be showed to the user. Wireless sensor network, have sensors which are connect to one node of the network. There are different types of topologies that can be used on that network, which the most common is the Mesh Network because, if one of the nodes stops working, the message is routed and sent through other nodes until reach principal node. Next step is send that information to the Cloud which, in the most used technics, the user has to have one gateway in one fixed point, where the Internet access has to be available or for other hand, if the Internet access is not available, the user needs to pay taxes to mobile phone operators, or one fixed point with electric installation to turn on the devices.

In this thesis, the propose solution is to use one mobile phone instead the conventional gateways, and with that, the user can put the sensor devices where he wants. These sensors can be uploaded and configured Over-The-Air by the application layer of the Synapse device. Sensors are ready after configuration, and they start, at that time, collet and send information through the Mesh network to the principal node. Principal node has one Synapse device connected to a microcontroller (Arduino) which has one SD card reader and a Bluetooth (BLE) device. All that collected information by the sensors is saved on the SD card until an user pass in front of principal node with a mobile

phone (Smartphone Android with our app installed) and with Bluetooth on. With all these requirements the Bluetooth and the mobile phone can be connected and the app send the newest date of one specific sensor on the Cloud database to the Arduino and after that, the Arduino delete the oldest dates present on the SD card. In response, the Arduino sends to mobile phone the oldest date collected by the sensor after delete, updating the cloud database.

This concept application can be used when does not exists a wireless or wired connection to the Internet or when one user want to put his sensors on the forest, and the nodes are going to send all collected information through the Mesh network. And the user has to choose one strategic place to put the principal node, which has to be in one place where many people pass in front of that node, to receive the information of the sensors on their mobile phones.

Keywords: Wireless Sensor Network, Synapse, ZigBee, IEEE 802.15.4, Bluetooth, BLE, Arduino, Domotics, Internet of Things, Gateway, Mobile Platforms, Cloud

Índice

1	Introdução	- 1 -
1.1	Enquadramento e Motivação	- 1 -
1.2	Identificação do problema.....	- 2 -
1.3	Objetivo	- 4 -
1.4	Projeto IrRADIARE	- 5 -
1.5	Organização da Dissertação	- 6 -
2	Estado da Arte	- 7 -
2.1	Tecnologias sem fios e tecnologias com fios	- 7 -
2.1.1	Vantagens e desvantagens	- 7 -
2.1.2	Topologias	- 9 -
2.1.3	Tipos de Comunicações sem Fios.....	- 11 -
2.2	Redes de Sensores Sem Fios.....	- 17 -
2.2.1	Arquitetura das WSN	- 19 -
2.2.2	Arquitetura ZigBee e Sinapse.....	- 21 -
2.2.3	Encaminhamento de mensagens.....	- 23 -
2.2.4	Segurança	- 25 -
2.2.5	Recolha de Energia (<i>Energy Harvesting</i>)	- 26 -
2.3	Sensores	- 28 -
2.3.1	Tipos de sensores	- 28 -
2.4	Plataformas para telemóveis.....	- 32 -
2.5	Serviços Web e <i>Cloud</i>	- 33 -
2.6	Projetos Similares	- 34 -
3	Proposta	- 39 -
3.1	Proposta para solução do problema.....	- 39 -
3.2	Arquitetura do modelo proposto	- 42 -
3.3	Proposta de criação de uma aplicação para PC	- 44 -
4	Validação.....	- 47 -
4.1	Escolha dos métodos de comunicação	- 47 -
4.1.1	Comunicação do nó principal com o <i>gateway</i>	- 47 -
4.1.2	Escolha do equipamento de comunicação da rede em Malha	- 49 -
4.1.3	Componentes utilizados	- 52 -
4.2	Comunicação através do Arduino.....	- 52 -
4.2.1	Informação recebida da rede em Malha.....	- 53 -

4.2.2	Informação enviada por Bluetooth para o telemóvel	- 54 -
4.3	Programas desenvolvidos em Python.....	- 58 -
4.3.1	Interface com o utilizador – Registo	- 58 -
4.3.2	Login e registo de sensores.....	- 59 -
4.3.3	Comunicação entre Arduino e Synapse	- 61 -
4.4	Consumo e especificações	- 63 -
4.4.1	Bluetooth	- 65 -
4.4.2	Synapse.....	- 67 -
4.4.3	Arduino UNO	- 69 -
4.5	Resultados da validação	- 71 -
4.5.1	Montagem do protótipo	- 71 -
4.5.2	Comunicação com o Serviço Web.....	- 72 -
4.5.3	Teste de operação	- 74 -
4.5.4	Teste de expansão	- 76 -
5	Conclusões e Trabalhos Futuros	- 79 -
5.1	Conclusões.....	- 79 -
5.2	Trabalhos Futuros	- 81 -
	Bibliografia	- 83 -
	Anexo	- 87 -

Índice de Figuras

Figura 2.1- Tipos de Topologias.....	- 10 -
Figura 2.2 - Comunicação sem fios.....	- 12 -
Figura 2.3 - Canais com separação entre portadoras	- 15 -
Figura 2.4 - Exemplo tipo quando um nó deixa de funcionar.....	- 17 -
Figura 2.5 - Arquitetura de acordo com o Modelo OSI.....	- 19 -
Figura 2.6 - Semelhanças do modelo OSI utilizadas no ZigBee.....	- 21 -
Figura 2.7 - Arquitetura Synapse.....	- 22 -
Figura 2.8 - Algoritmo AODVjr.....	- 23 -
Figura 2.9 - Algoritmo utilizado pelo Synapse	- 24 -
Figura 2.10 - Modelo comum de Recolha de Energia.....	- 27 -
Figura 2.11 - Modelo de Recolha de Energia.....	- 27 -
Figura 2.12 - Evolução na utilização da Rede de Sensores	- 28 -
Figura 2.13 - Tipos de Sensores (Termopar, RTD, Termístores respetivamente)	- 30 -
Figura 2.14 - Sensor UV	- 31 -
Figura 2.15 - Sensor de Monóxido de Carbono	- 31 -
Figura 2.16 - Conceito <i>Coud</i>	- 34 -
Figura 2.17 - Produto Valarm com a utilização de um telemóvel como <i>gateway</i>	- 35 -
Figura 2.18 - SensorCloud com um <i>gateway</i> (necessário ligação à internet)	- 36 -
Figura 2.19 - Libelium com um <i>gateway</i> ligado por cabo ou sem fios (necessário ligação à internet)	- 37 -
Figura 3.1 – Comunicação entre a rede, dispositivo e sensor	- 39 -
Figura 3.2 - Comunicação entre a rede, dispositivo de comunicação e <i>gateway</i>	- 41 -
Figura 3.3 - Arquitetura	- 43 -
Figura 4.1 - Adaptadores para o Synapse	- 52 -
Figura 4.2 - Shield para Arduino, Synapse e Cartão SD.....	- 52 -
Figura 4.3 - Fluxograma visto do Arduino.....	- 57 -
Figura 4.4 - Registo de um novo utilizador	- 58 -
Figura 4.5 - Pedido a um serviço web.....	- 59 -
Figura 4.6 - Bluetooth TinySine	- 65 -
Figura 4.7 - Synapse RF200P81.....	- 67 -
Figura 4.8 – Módulo para ATmega328P	- 70 -

Figura 4.9 - Protótipo	- 71 -
Figura 4.10 - Exemplo de um registo de um novo utilizador	- 72 -
Figura 4.11 - Web Site criado na dissertação Ponto de Acesso Móvel em Ambiente Sensorial .	- 72 -
Figura 4.12 - Diferenças de um utilizador particular (lado esquerdo) e de uma organização (do lado direito).....	- 73 -
Figura 4.13 - Exemplo da adição de um novo sensor	- 73 -
Figura 4.14 - Ficheiro .txt que representa os valores recebidos da rede de sensores	- 74 -
Figura 4.15 - Exemplo da atualização do cartão SD (envio de dados na aplicação Android)	- 75 -
Figura 4.16 - Exemplo da receção de dados enviados pelo Bluetooth	- 76 -
Figura 4.17 – Diretorias criadas no cartão SD com formato YYYY/MM/DD/HH	- 77 -
Figura A.1 – Tabela de comparação de dispositivos Synapse	- 87 -

Índice de Tabelas

Tabela 2.1 – Frequências típicas da camada física	- 21 -
Tabela 2.2 - Comparação da utilização de sistemas operativos nos Smartphones.....	- 32 -
Tabela 4.1 - Tabela de comparação de tecnologias para <i>gateway</i> -Arduino (1)	- 48 -
Tabela 4.2 - Tabela de comparação de tecnologias para <i>gateway</i> -Arduino (2)	- 48 -
Tabela 4.3 - Tabela de comparação de tecnologias para a rede em Malha	- 50 -

Acrónimos

AES	Advanced Encryption Standard
AODV	Ad hoc On-Demand distance Vector
BLE	Bluetooth Low Energy
CDMA	Code Division Multiple Access
CSMA	Carrier Sense Multiple Access Collision Avoidance
DYMO	Dynamic MANET On-demand
GSM	Global System for Mobile communication
HSPA	High Speed Packet Access
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
JSON	JavaScript Object Notation
MAC	Media Access Control
OSI	Open Systems Interconnection
RF	Radio Frequency
RPC	Remote Procedure Call
RTD	Resistive Temperature Detector
SDK	Software Development Kit
SNAP	Synapse Network Application Protocol
SSL	Secure Sockets Layer
TKIP	Temporal Key Integrity Protocol
WDSL	Web Services Description Language
WEP	Wired Equivalent Privacy
WiMAX	Worldwide Interoperability for Microwave Access
WLAN	Wireless Local Area Network
WMAN	Wireless Metropolitan Area Network
WPA	Wi-Fi Protected Access
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network
WWAN	Wireless Wide Area Network

1

Introdução

1.1 Enquadramento e Motivação

Cada vez mais a Internet das Coisas (*Internet of Things* – IoT) é mais utilizada nos dias de hoje. [1] A população quer ter os seus *gadgets*, eletrodomésticos, casas, carros, etc. ligados à internet de modo a obter a mais vasta informação dos mesmos, controlando melhor o seu quotidiano e tendo acesso a toda essa informação em qualquer lado e instante.

Para que tal seja possível, na maior parte dos casos, é necessário uma rede de sensores sem fios (*Wireless Sensors Network* – WSN) que recolhe toda a informação necessária e útil para o utilizador, que posteriormente, com uma ligação à internet, a informação ficará disponibilizada na *Cloud*, mais propriamente, num servidor que guardará os dados referentes aos sensores.

As WSN são uma área em pesquisa que tem tido um rápido e desejado crescimento, de modo a tornarem-se cada vez mais uma tecnologia de baixo custo e escalável. Muitos protocolos e algoritmos foram propostos para a rede de sensores [1] e cada vez são mais as áreas de interesse na sua utilização nas mais diversas aplicações, como por exemplo: [2]

- Saúde – controlo de administração de medicamentos no hospital, monitorização dos médicos e dos doentes;
- Militar – monitorizar forças amigas, reconhecimento de terreno inimigo, reconhecimento de ataque nuclear, biológico ou químico;
- Segurança – intrusão, deteção de movimento, perseguição;

- Ambiente – deteção de fogo, chuva, temperatura, deteção de inundação, humidade, qualidade do ar;
- Habitações – ambientes e casas inteligentes, temperatura de cada divisão da casa, controlo de eletricidade e fluxo de água.

Os sensores da rede podem ser constituídos por diversos tipos, como por exemplo, de temperatura, humidade, movimento, pressão, intensidade de luz, ruído, deteção de presença de um determinado objeto, deteção da velocidade, direção/tamanho/aceleração de um objeto, etc..

Esta dissertação foi realizada com a parceria da empresa IrRADIARE. A IrRADIARE é uma empresa que trabalha mais para a vertente da monitorização urbana, redução de custos energéticos, otimização no processo de segurança, proteção do ambiente, entre outras.

O projeto foi dividido em duas partes:

- a parte de mais baixo nível – incluindo os sensores, a rede de sensores e a comunicação com um *gateway* (neste caso um dispositivo móvel);
- a parte de mais alto nível – incluindo o *gateway*, web site e toda a parte envolvente do processamento dos dados na internet, feita pelo colega José Ricardo de Lima Abrantes, Ponto de Acesso Móvel em Ambiente Sensorial.

1.2 Identificação do problema

A tecnologia tem vindo a tornar-se uma dependência e tem criado uma melhor qualidade de vida para vários utilizadores. A utilização da Internet nos mais diversos equipamentos tem tido uma vasta procura, nomeadamente em aparelhos elétricos ou eletrónicos, permitindo que o utilizador possa ter acesso a toda informação dos mesmos em qualquer lugar e instante. É neste tópico que foi desenvolvida esta dissertação – Sensores “em movimento”.

A utilização desses sensores não passa só pelo entretenimento do utilizador, ou seja, não está relacionado somente em controlar a que a temperatura está a sua casa regularmente, ver o número de vezes que entra e sai de casa, etc.. Existem muitas outras aplicações que, ao serem utilizados os sensores, podem ajudar o meio em questão, como por exemplo, na deteção de um incêndio ou no controlo da qualidade do ar.

Foram propostas várias soluções de conseguir obter informação através de sensores, e torná-la acessível em qualquer local, disponível a qualquer momento, sendo que todas elas possuem seus prós e contras.

Uma das propostas frequentemente utilizada é a aplicação de uma rede em Malha (*Mesh*) de sensores permitindo que comuniquem e recebam todos os dados necessários para posteriormente serem entregues ao sensor principal/coordenador. Posteriormente existem algumas formas de os dados serem disponibilizados na internet, ou mais concretamente na *Cloud*. A solução mais utilizada é uma ligação direta à rede, por cabo *Ethernet*, em que os dados são recebidos, tratados e enviados para a *Cloud*, ficando posteriormente disponíveis ao utilizador. Este método torna-se um problema quando os edifícios ou locais, onde será necessária a utilização destes equipamentos, não possuem uma ligação *Ethernet* próxima. Sendo assim a melhor solução a utilização de redes sem fios (*Wireless*). Fica assim possível a instalação de sensores noutros locais (menos acessíveis), permitindo a receção dos dados na *Cloud* assim que a ligação a um Encaminhador (*Router*) esteja estabelecida.

Volta a existir um outro problema no caso de existir um corte de corrente elétrica no edifício onde o Encaminhador estava instalado, assim, os dados que supostamente eram para ser recebidos, são perdidos se o sensor coordenador não possuir nenhuma forma de corrigir este problema de ligação. Por exemplo, a colocação de um dispositivo que proceda ao armazenamento dos dados (juntamente com uma bateria auxiliar para estes continuarem em funcionamento) e posteriormente que envie os dados em falta.

Existe ainda um outro problema quando nenhum dos equipamentos, anteriormente referidos, estão disponíveis, ou seja, tanto a ligação *Ethernet* bem como a ligação sem fios (Wi-Fi) a um Encaminhador não estão diretamente acessíveis, como é o caso de uma herdade com grande dimensão, ou num parque onde se pretende avaliar a quantidade de humidade presente no solo, permitindo saber se será mesmo necessário proceder à rega. Para corrigir este problema, pode existir a possibilidade da colocação de um cartão que estabeleça uma ligação 3G ou 4G (formas de comunicação móveis), garantindo assim a comunicação com a internet de forma a atualizar os dados na *Cloud*. Isto implica que, por cada sensor coordenador instalado, o utilizador estará dependente das tarifas que a operadora lhe cobrar por cada cartão.

Revendo todos os factos atrás referenciados, verificou-se que existe um problema de como se pode garantir que os dados dos sensores instalados pelo utilizador cheguem ao seu destino de forma económica e de fácil implementação.

1.3 Objetivo

Existe um problema de fazer chegar os dados ao utilizador de forma económica e de fácil implementação. O objectivo desta dissertação é encontrar um *gateway* (ponto de acesso) entre os sensores e a *Cloud* que permita que o próprio receba dados de sensores e posteriormente os disponibilize ao utilizador, sem que tenha como requisito principal, um ponto fixo que internet ou alimentação.

Os sensores estão ligados numa rede em Malha (capítulo 2.1.2) incluindo Dispositivos Finais, Encaminhadores e Coordenadores. Assim a informação é passada dos Dispositivos Finais para os Encaminhadores que irão escolher o melhor caminho para enviar toda essa informação para o Coordenador. Estes módulos têm uma bateria acoplada para que o posicionamento dos mesmos não esteja limitado, ficando independentes de uma instalação elétrica. A escolha destes módulos devem de cumprir alguns requisitos como baixo custo e baixo consumo de energia, de modo a que o utilizador não tenha que mudar frequentemente as baterias dos dispositivos. Irá ser utilizado o protocolo IEEE 802.15.4 para a comunicação entre os dispositivos pois este protocolo está bastante otimizado e cumpre os requisitos necessários. Os dispositivos têm que cumprir requisitos, como é o caso do consumo, alcance e baixo custo, pois é pretendido que o utilizador desta rede de sensores possa usufruir da mesma sem grandes intervenções.

Depois de escolhido como os sensores se irão interligar, o desafio é enviar os dados dos sensores para a *Cloud*. O Coordenador comunicará através de um *gateway* enviando uma informação de forma a saber se o mesmo permite receber dados recolhidos pelos sensores que estão no local onde se encontra. Assim, quando estabelecida a ligação entre o *gateway* e o dispositivo principal, são enviadas pequenas quantidades de dados e que quando este estiver conectado à internet enviará os dados recebidos para a *Cloud*.

Mas como se terá garantia que não se perderá informação? E se o *gateway* naquele momento recusar a receção dos dados? E se ninguém se aproximar do sensor durante muito tempo, a informação será perdida? E como tornar esse *gateway* um ponto não fixo?

Este será o objetivo deste projeto.

1.4 Projeto IrRADIARE

Os sensores estão diretamente ligados por cabos a dispositivos que regularmente recebem informação dos mesmos, sendo esses dispositivos módulos de comunicação sem fios. Essa informação passa de dispositivo para dispositivo dentro da rede de sensores até chegar ao dispositivo principal. Esse dispositivo é responsável por agrupar todas essas mensagens e armazená-las. Este tem acoplado um dispositivo de comunicação, que servirá para comunicar com o *gateway*.

Quando um utilizador se aproxima do dispositivo principal, com um *gateway*, este envia um pedido de conexão, caso o utilizador assim o permita e tenha a aplicação necessária instalada no seu *gateway*, a ligação é estabelecida. Essa aplicação, envia uma mensagem com a data mais recente contida na base de dados e, assim, o dispositivo principal procura todos os dados, anteriores a essa data, existentes no cartão e elimina-os (uma vez que já se encontram na base de dados na internet). De seguida, o dispositivo principal, procura no armazenamento interno, a data mais antiga para proceder ao envio dos dados para o *gateway*. Após o envio, desliga a conexão e retoma o processo anterior de recolha de dados dos sensores.

O *gateway*, caso não tenha uma ligação à internet, é armazenado na sua capacidade interna, o ficheiro completo com os dados recolhidos (estes dados não ficam acessíveis ao utilizador). Quando existir uma ligação à internet, o *gateway* envia os dados para uma aplicação que está em funcionamento no servidor, usando uma tecnologia que vai tratar os dados e verificar na base de dados se essa informação já existe (os dados são enviados várias vezes para diferentes *gateways* e para a aplicação no servidor, até se garantir que esses mesmos dados já se encontram no servidor, sendo essa confirmação, a mensagem enviada para o dispositivo principal com a última data encontrada no servidor). Por fim os dados ficam acessíveis ao utilizador, através de uma página de internet, esses são recolhidos por serviços da base de dados anteriormente referida.

Com o desenvolver de todo o projeto demonstrado nesta dissertação o maior contributo está relacionado com a facilidade que o utilizador poderá ter em colocar os seus sensores, sendo que:

- O utilizador não fica dependente de pontos fixos ou pontos onde seja necessário ter cobertura de internet;
- O conceito pode ser usado por exemplo numa floresta, podendo prevenir incêndios;

- De acordo com a empresa IrRADIARE a sua utilização poderia vir a ser feita para substituição de grandes centrais de sensores da qualidade do ar, ou usado em centros históricos, pois não é necessário mais nenhum recurso para além dos sensores.

1.5 Organização da Dissertação

Esta dissertação está dividida em seis capítulos, sendo eles:

- Introdução – É demonstrado qual o problema existente aquando de uma utilização de uma rede de sensores e um *gateway*, referindo o objetivo deste projeto e uma breve solução. É também aqui descrito, de forma resumida, o funcionamento de todo o projeto (incluído nas duas dissertações);
- Estado da Arte – É dado a conhecer todos os termos e pesquisas presentes nesta dissertação. É falado de todas as escolhas que estiveram em causa a sua utilização, bem como o fundamento teórico do funcionamento das escolhas principais;
- Proposta – Demonstração da criação do projeto, referindo como vai funcionar cada dispositivo. Contém a arquitetura da proposta bem como os diagramas de comunicação entre dispositivos;
- Validação – Descrição e justificação das escolhas dos equipamentos para o projeto. Para facilitar a compreensão estas escolhas foram feitas e demonstradas através de tabelas. Consumos teóricos, de modo a dar a previsão da duração das baterias. E, por fim apresentação de todo o conceito e justificações para os erros obtidos durante a realização do projeto. Os resultados são apresentados através de imagens;
- Conclusões e Trabalhos Futuros – Apresentação das conclusões com recomendações a serem realizadas no projeto, pois para ser um produto final é necessário algumas alterações, de forma a torna-lo mais eficaz e eficiente;
- Bibliografia – Referências bibliográficas.

2

Estado da Arte

2.1 Tecnologias sem fios e tecnologias com fios

Existe cada vez mais impacto na evolução e na utilização de equipamentos sem fios. Mas quais serão as suas vantagens e desvantagens? Quão vantajoso é para prescindir de uma ligação com fios?

2.1.1 Vantagens e desvantagens

Assumindo um exemplo comum: uma ligação de dois computadores à mesma rede (LAN – *Local Access Network* – Rede de Acesso Local). Se a ligação dos dois for estabelecida por cabo, geralmente é necessário um *hub*, *switch* ou encaminhador e dois cabos *Ethernet* para poderem partilhar a mesma ligação, tendo que os dois computadores permanecerem fixos num único lugar. Se a ligação for feita sem fios com tecnologia Wi-Fi (por exemplo), os utilizadores podem colocar os seus computadores em qualquer outro local (limitado pelo alcance dos mesmos).

Um caso mais genérico, por exemplo, nas tecnologias com fios, esta ligação pode ser feita por um *hub* ou *switch* (caso se pretenda ter mais do que um dispositivo ligado à mesma rede pois, por vezes, o encaminhador não tem terminais suficientes para mais do que um dispositivo) onde anteriormente estará ligado a um encaminhador para permitir que esse mesmo dispositivo tenha acesso à internet. Nestas tecnologias, na sua maioria, os componentes pelos quais são compostos são mais baratos do que comparativamente com as redes sem fios, bem como um baixo consumo

de energia, pois existe muita potência dissipada pelas antenas multidirecionais nas redes sem fios. A ligação com fios é também mais segura, visto que não existe transmissão de dados pelo ar, o que dificulta a sua interceção por piratas informáticos (*hackers*). No caso da ocorrência de um problema na ligação, com esta tecnologia é mais fácil a sua deteção, pois se existir uma ligação de A para C passando por B, e se A e B tiverem acesso à internet e C não, facilmente se pode concluir que o problema estará no cabo que os une ou possivelmente o dispositivo C poderá não estar nas melhores condições.

Algumas das limitações das tecnologias com fios está relacionada com a ligação física entre dois dispositivos, o que dificulta a deslocação/alteração do local onde esses dispositivos possam ser colocados, visto que, frequentemente, os cabos, *switchs* e encaminhadores têm que estar fixos. Existe também um outro problema que corresponde à potência dissipada nos cabos, caso a distância da ligação de um dispositivo para outro seja significativamente grande.

As tecnologias sem fios têm uma maior flexibilidade na instalação dos equipamentos, o que permite ter um maior alcance, onde possivelmente os dispositivos que necessitam de uma rede com fios não poderiam ser instalados. Isto evita a passagem de cabo e calhas na parede. Ainda assim esta tecnologia está dependente de um único ponto com ligação com fio, sendo que é necessário que um dos encaminhadores tenha uma ligação direta à internet. Após esta ligação todos os outros dispositivos, desde que estejam ao alcance da rede, podem conectar-se à internet. Os componentes deste tipo de tecnologias são um pouco mais caros e consomem mais energia do que nas redes com fios. As ondas de rádio emitidas por estes equipamentos, por trabalharem na mesma banda de frequência que outros equipamentos, como o Bluetooth, podem fazer com que por vezes ocorram interferências com os mesmos. Neste tipo de ligação, a segurança e a qualidade de serviço por vezes é relativamente baixa pois, as ligações podem ser intercetadas por piratas informáticos. Existe ainda um outro problema, pois esta tecnologia não contorna obstáculos que, no caso de existirem paredes sucessivas, pode fazer com que a informação não chegue ao seu destino ou a intensidade do sinal seja baixa.

Para tal é necessário reconhecer quais os desafios das WSN para garantir que nada falha no sistema, como o tipo de serviço, qualidade de serviço, tolerância a falhas, tempo de vida, escalável, vasta gama de densidades, programável e manutenção.

2.1.2 Topologias

As topologias são normalmente esquemas que definem como é que as ligações das redes vão ser executadas, incluindo os nós e as ligações entre eles. As topologias, como pode ser observado na figura 2.1, podem ser caracterizadas pelos seguintes tipos [3]:

- Barramento (Bus) – é constituída por uma única ligação, em que todos os dispositivos estão ligados à mesma. Neste tipo de topologia quando um dispositivo tenta comunicar com outro, este envia uma mensagem *Broadcast* para a única ligação (cabo) existente, em que todos os dispositivos a ele ligado vão conseguir ver a mensagem, mas somente o destinatário a aceitará e a processará. Como é de esperar esta ligação é bastante fácil de utilizar mas tem bastantes problemas, como é o caso de, por exemplo, uma ligação que seja de A para B e posteriormente para C, se a ligação para B não estiver operacional e a mensagem for direcionada de A para C, então C nunca irá receber essa mensagem;
- Anel – cada dispositivo tem dois vizinhos, pois se um deles deixar de funcionar, a mensagem pode ser enviada pelo outro dispositivo vizinho até ao seu destino. Esta topologia tem também alguns problemas em termos consistência pois basta uma falha num dispositivo para que a rede deixar de funcionar corretamente;
- Estrela – existe um dispositivo central que troca informação com todos os dispositivos à sua volta, como é o caso de um *switch* ou encaminhador. Caso um dos dispositivos ligados ao dispositivo central falhe, não irá interferir no funcionamento da rede, pois são dispositivos independentes;
- Árvore – várias topologias em estela ligadas em Barramento;
- Malha – nesta topologia cada dispositivo tem um algoritmo que reencaminha todas as mensagens para o dispositivo que estiver: mais próximo, com mais fiabilidade ou com o caminho mais favorável; isto de acordo com o algoritmo pré-estabelecido. Esta topologia é importante neste projeto pois quando um dispositivo necessita enviar a sua informação até ao dispositivo principal e existe algum problema na ligação mais direta então, através do algoritmo estabelecido, é fornecida uma outra rota/forma de essa informação chegar corretamente ao dispositivo principal sem comprometer a receção da informação. Será esta a topologia utilizada nesta dissertação, pois é aquela que melhor satisfaz todas as necessidades de fluxo de mensagens.

Mais especificamente as redes em Malha pertencem ao grupo de trabalho do IEEE 802.15 e para se perceber como funcionam este tipo de redes tem que se perceber o quão diferente são das redes

tipicamente usadas para ligar aparelhos como telemóveis e computadores pessoais. Nessas redes existe apenas um único ponto (HUB) que recebe todos os dados e faz de Ponto de Acesso para os dispositivos.

Nas redes em Malha os dispositivos estão ligados a encaminhadores, normalmente denominados por nós, que interligam todos os dispositivos e enviam os seus dados de nó para nó até chegar ao seu destino final. Mas o que são estes nós e como funcionam?

Os nós permitem que os dados enviados pela rede sejam novamente enviados por esse mesmo nó para um outro, ou seja, os nós são capazes de receber e enviar dados sempre que necessário. Estes não são escolhidos aleatoriamente, têm um algoritmo (podem existir muitos algoritmos diferentes, dependendo da finalidade, fluidez e garantia que se desejar) que determina qual será o próximo nó para onde terá que enviar os dados. São estes mesmos algoritmos que também detetam quando um nó deixa de estar disponível ou simplesmente deixar de existir, e assim gerar um novo caminho, de forma que os dados cheguem ao seu destino sem problemas. Quando se instala um novo nó, a rede em Malha vai reconhecer que existe um outro caminho que pode seguir e vai integrá-lo na rede.

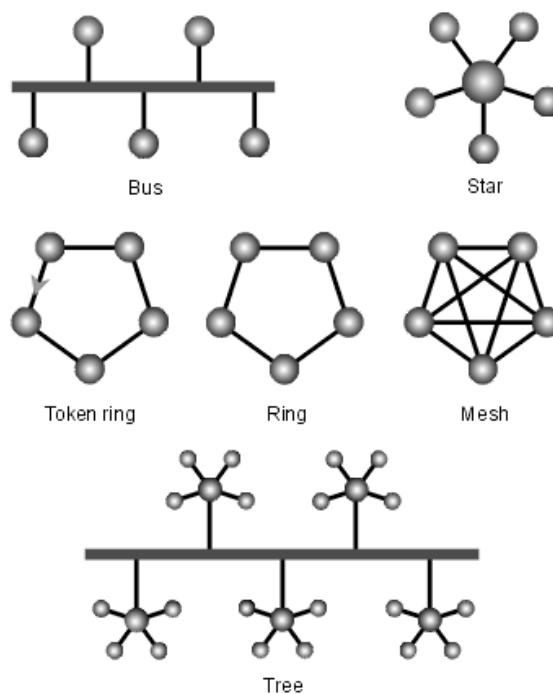


Figura 2.1- Tipos de Topologias [3]

2.1.3 Tipos de Comunicações sem Fios

Para um melhor fundamento na escolha da tecnologia utilizada neste projeto, serão apresentadas algumas formas de comunicação, tanto para a comunicação entre sensores como entre o dispositivo móvel e o nó principal.

A evolução da tecnologia deu origem ao aparecimento de várias alternativas e protocolos que permitem a transmissão de dados sem fios. Desde então existem quatro grandes grupos responsáveis pelos *standards* IEEE (Instituto dos Engenheiros Elétricos e Eletrónicos – *Institute of Electrical and Electronics Engineers*). Esta é uma organização profissional sem fins lucrativos, promovendo o desenvolvimento tecnológico para o bem da sociedade [4].

O IEEE, como pode ser verificado na figura 2.2, divide as redes sem fios em:

- WPAN (Rede de Área Pessoal Sem Fios – *Wireless Personal Area Network* (802.15)) – São destinadas para sensores/redes sem fios de curto alcance. Tivera início com o Bluetooth (802.15.1) que teve um grande impacto no envio de dados através dos telemóveis tornando-se bastante popular. Além de ser uma tecnologia de baixo custo, também permitiu a comunicação entre computadores, telemóveis e outros dispositivos móveis. O protocolo 802.15.2 é um outro subgrupo que trata, principalmente, das interferências criadas pela frequência de trabalho 2.4GHz do WLAN e WPAN. Já o protocolo 802.15.4 tem como preocupação o estudo de versões de baixo custo e de baixa taxa de transferência de dados, o qual é bastante usado nas redes em Malha para controlo de sensores, permitindo assim um consumo reduzido, levando as baterias a durarem meses ou anos. [5] [6]
- WLAN (Rede de Área Local Sem Fios – *Wireless Local Area Network* (802.11)) – Este grupo teve e continua a ter um papel importante na evolução das tecnologias sem fios, possibilitando a interconectividade entre as redes WPAN e WLAN. Cada dispositivo que se conecta a uma WLAN pode ser dividido em duas categorias: Ponto de Acesso e Cliente. Por exemplo, os encaminhadores, são um Ponto de Acesso, sendo o seu principal objetivo receber e reenviar o sinal, enquanto que o Cliente pode ser um portátil, um telemóvel, impressora, etc. [7]. Um dos principais utilizadores deste grupo é o Wi-Fi.
- WMAN (Rede de Área Metropolitana Sem Fios – *Wireless Metropolitan Area Network* (802.16)) – Tal como nas redes WLAN existe um principal utilizador das redes WMAN, que é o caso do WiMAX. Este grupo é responsável por fornecer uma ligação para uso doméstico ou empresarial através de um único Ponto de Acesso. [5] [6]

- WWAN (Rede de Área Mundial Sem Fios – *Wireless Wide Area Network*) – Este grupo tem como foco principal a rede sem fios a longas distâncias, permitindo uma ligação regional, nacional ou mundial. Este grupo é utilizado pelas operadoras móveis nas suas redes de transmissão (GSM, CDMA, HSPA, etc.). [5] [6]

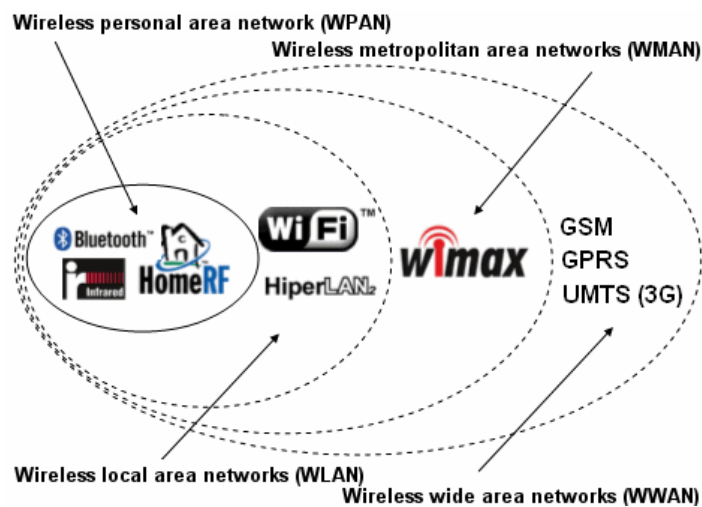


Figura 2.2 - Comunicação sem fios [8]

As redes WPAN e WLAN são as que têm mais ênfase neste projeto. Como tal existem muitas formas de comunicação e bastantes equipamentos/protocolos que podem satisfazer os requisitos pedidos. De seguida será descrito a forma de trabalho e protocolos utilizados.

BLUETOOTH

O Bluetooth é uma tecnologia sem fios de baixo custo, baixo consumo e de curto alcance usada para transferir dados entre diferentes tipos de dispositivos móveis e redes LAN. Hoje em dia, o Bluetooth é utilizado na sua grande maioria para a comunicação entre dispositivos eletrónicos.

Criado por um grupo de engenheiros da Ericsson em 1994, acabando por se tornar oficial em 1998, para ser uma alternativa sem fios aos cabos RS-232 (portas série), este opera numa banda não industrial, científica e médica de 2.4GHz a 2.485GHz. A partir de então o desenvolvimento da tecnologia permitiu ligação sem fios para chamadas de voz (mãos livres), impressoras /fax, relógios inteligentes, sincronização automática com os PDA, teclados e ratos, entre outros. O Bluetooth permite também a partilha de fotos, musica, vídeos, dados e voz, etc..

Desde 1998, até aos dias de hoje, a tecnologia Bluetooth tem vindo a evoluir, já contando com quatro versões, sendo uma delas utilizada no desenvolvimento desta dissertação – Bluetooth 4.0 BLE (Bluetooth de Baixo Consumo – *Bluetooth Low Energy*). Bluetooth BLE é uma tecnologia

desenvolvida para consumir apenas uma fração de potência, quando comparado com outras tecnologias Bluetooth. [9]

Esta tecnologia trabalha tipicamente com um dispositivo mestre (*master*) e um ou mais dispositivos escravos (*slaves*), e a comunicação não necessita de ser direta entre dispositivos. Se um obstáculo estiver entre eles ou caso não estejam perfeitamente alinhados, como é o caso da tecnologia Infravermelho [10], não existirá perda de informação.

WI-FI

O Wi-Fi é um *standard* utilizado nas comunicações com o protocolo 802.11 (WLAN). Este veio facilitar a comunicação através da internet, eliminando passagens de cabo, caso o utilizador quisesse alterar a posição do seu computador. Através desta tecnologia foi possível conectar vários dispositivos na mesma rede, como por exemplo, telemóveis, tablets, impressoras, máquinas de lavar a louça ou roupa, etc.. Existem cinco versões de comunicação [11]:

- 802.11a
- 802.11b
- 802.11g
- 802.11n
- 802.11ac

A comunicação 802.11a é a mais antiga e mais cara, o que levou a comunicação 802.11b a ser mais utilizada. A sua velocidade de transmissão chegava no máximo até 11Mb/s e podia ter um alcance de 400m em áreas abertas ou 50m em espaços fechados. À medida que a tecnologia foi evoluindo, também a velocidade de transmissão foi aumentando. Na comunicação 802.11ac, a velocidade de transmissão é de 1.3Gb/s.

Apesar de esta tecnologia ser de fácil utilização e instalação, normalmente é mais cara que as opções de ligação por cabo e, o seu rendimento, é inversamente proporcional à distância entre os dispositivos e o ponto de acesso, ou seja, quanto mais afastado estiver menos “força” de sinal terá.

Dentro desta tecnologia, desenvolveu-se uma aplicação denominada Wi-Fi Direct. Esta é uma comunicação entre dispositivos em que não é necessário acesso à internet, ou seja, basta que dois dispositivos possuam uma placa Wi-Fi com essa tecnologia, para poderem partilhar informação.

NFC

A comunicação NFC (Campo de Comunicação Próximo – *Near-Field Communication*) tem sido uma grande aposta utilizada, principalmente nos telemóveis. NFC é um *standard* definido pelo Fórum NFC, um consórcio global de hardware, software, companhias de cartões de crédito, bancos, provedores de internet, e outras instituições, com o intuito de desenvolver esta tecnologia. [12]

A comunicação é de muito curto alcance e foi desenhada com o intuito de unir os telemóveis e os cartões de crédito. Esta funciona quando dois dispositivos com NFC (ou um dispositivo e um chip (TAG)) se tocam ou quando estão relativamente perto, cerca de 30cm [13] e é estabelecida a ligação.

A utilidade do NFC passa por substituir, principalmente cartões de crédito, bilhetes de avião, transportes públicos, leitura de uma *Tag* RFID (Identificação por Rádio-Frequência – *Radio-Frequency IDentification*), etc., como por exemplo, obter a informação de um eletrodoméstico, aproximando um telemóvel ao mesmo. [14]

Infravermelhos

A comunicação por infravermelhos é uma comunicação sem fios e ainda é bastante utilizada, como por exemplo, nos comandos das televisões (os telemóveis mais recentes ainda utilizam esta tecnologia, para substituir o comando), no controlo de carros telecomandados, e foi usada durante algum tempo na transmissão de dados entre telemóveis. Esta tecnologia foi suprimida pelo Bluetooth, pois a comunicação entre dispositivos não é omnidirecional e o seu alcance é relativamente baixo, que em casos bastante favoráveis pode chegar aos 10 metros. A sua intensidade é alterada quando existe

algum objeto entre os dois dispositivos, ou seja, é necessário que o emissor e o recetor estejam praticamente alinhados, sem nenhuma perturbação pelo meio, para garantir que a comunicação é realizada com sucesso. [10] [15]

ZigBee e Sinapse

Estes dois dispositivos de comunicação (ZigBee e Sinapse) são dois exemplos que usam o mesmo tipo de protocolo de comunicação, o IEEE 802.15.4 (pertencente as WPAN). E são muito utilizados em redes de sensores sem fios.

O ZigBee é uma tecnologia sem fios de baixo custo/consumo e funciona nas larguras de banda de 2.4GHz, (“podendo nesta frequência utilizar até 16 canais com uma separação entre portadoras de 5MHz” [16], como mostra a Figura 2.3), 900MHz e 686MHz. Este foi desenvolvido pela ZigBee Alliance que teve como objetivo interligar vários objetos, como lâmpadas, equipamentos de segurança, termostatos e termómetros, entre muitos outros que se podem encontrar numa habitação. É também utilizado em aplicações onde a velocidade de transmissão de dados não seja muito necessária, mantendo uma longa durabilidade da bateria acoplada ao mesmo.

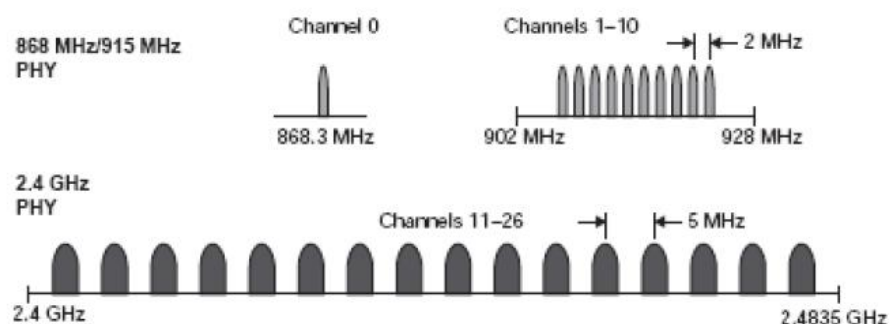


Figura 2.3 - Canais com separação entre portadoras [16]

O ZigBee tem como principais características o baixo consumo, baixo custo e de pequenas dimensões. Devido às suas baixas potências e período de transmissão, o dispositivo utilizado pode durar durante alguns anos com duas baterias tipo AA sem necessitar de substituição (através dos modos Ativo e Adormecido (*Sleep*), pelo que consegue ter uma eficiência elevada). É constituído por dispositivos finais, encaminhadores e coordenadores e utiliza o protocolo IEEE 802.15.4, para tirar partido das camadas Física, MAC, da

frequência, da largura de banda e da técnica de modulação [17]. Esses dispositivos podem ser agrupados em duas classes:

- FFD (Dispositivos com Todas as Funções – *Full Function Devices*) que estão permanentemente ativos e prontos para receber dados, geralmente estão ligados a uma fonte de energia;
- RFD (Dispositivos de Funções Reduzidas – *Reduced Function Devices*) que ao contrário da FFD os dispositivos passam a maior parte do tempo adormecidos para poupar a energia das baterias. [18]

O dispositivo tem integrado um módulo RF (Rádio-Frequência – *Radio Frequency*) que normalmente é de curto alcance, mas como o ZigBee funciona com rede em Malha, permite cobrir uma vasta área com os dispositivos. Este está estruturado em camadas seguindo o Modelo OSI (Intercomunicação de Sistemas Aberto – *Open Systems Interconnection*). Para uso comercial do ZigBee, o custo pode ser bastante elevado.

O Synapse proporciona uma plataforma de baixo custo e longo alcance. Pode ser utilizado monitorização nas mais diversas aplicações e utilidades, tendo como especificações e benefícios a utilização de rede em Malha com auto formação (*multi-hop*), comunicação ponto a ponto, não é necessário a utilização de um coordenador, é programável através de um sistema operativo de rede sem fios em Malha denominado SNAP (baseado em Python) e possui um microcontrolador independente, que pode ser programado *Over-The-Air* (atualização sem necessitar de intervenção física diretamente no equipamento).

Este tem a facilidade de se poder adicionar um novo nó à rede sem que exista grandes problemas pois, lida perfeitamente com a complexidade da rede em malha, bastando indicar qual o canal em que se pretende que seja adicionado o novo nó e fica operacional. Tem facilidade de se poder fazer testes rápidos, como já dito anteriormente, através da atualização *Over-The-Air*, enviando novos scripts para o nó pretendido ou para todos os nós da rede. Tal como no ZigBee, o Synapse também segue as camadas do modelo OSI. [19] Através da figura 2.4, pode ser verificado o funcionamento da rede em Malha no estado normal, ou quando um dos dispositivos falha.



Figura 2.4 - Exemplo tipo quando um nó deixa de funcionar [19]

O SNAP tem a desvantagem de possuir pouca informação em relação à sua utilização (pouco feedback).

Outros tipos de comunicações

Para além das tecnologias acima apresentadas, existem muitas outras (que não vão ser aprofundadas) que podem ser utilizadas com o mesmo intuito, como é o caso da Z-Wave da Z-Wave Alliance, Libelium, Mi-Wi, iDom, X10, UPB, INSTEON. [20]

2.2 Redes de Sensores Sem Fios

As Redes de Sensores Sem Fios, ou WSN, como o nome indica são um conjunto de sensores que estão ligados em rede. Estes podem, ou não, comunicar entre si, dependendo da topologia aplicada a cada conceito, sendo a mais falada neste projeto a topologia em Malha. Esta permite que os sensores comunicam entre si, enviando a informação recolhida pelo caminho mais próximo até ao sensor principal/coordenador. Este tipo de tecnologia com o passar do tempo, com a utilização pedida e pela investigação exercida, a tendência será que se torne cada vez mais barata e cada vez mais pequena.

As WSN podem ter várias topologias, de acordo com o desejado, e cada nó possui um microcontrolador e/ou um microprocessador, um transmissor rádio (RF) e um ou mais sensores de modo a proceder à recolha dos dados. Estes podem ser utilizados para fins medicinais, militares, ambientais. O dispositivo, depois de obter a informação recolhida pelos sensores, poderá processar essa informação de forma que seja só enviado o necessário para não existir um congestionamento de mensagens na receção, ou ocupar muita largura de banda. Assim, pode ser enviado separadamente as mensagens caso não seja necessário o envio constante de informação.

Posteriormente essa informação passará por vários nós adjacentes, pertencentes à rede, até chegar ao nó principal. Este processo é repetido sempre que existir novos dados recolhidos pelos sensores.

As WSN são bastante diferentes dos outros tipos de redes, principalmente pelas suas aplicações, equipamento, interações em diferentes ambientes, possibilidade de expansibilidade, energia, etc..

Estas têm de cumprir certos requisitos [21] [22]:

- Tolerância a falhas – Os nós são propícios a falhas quer pelo ambiente onde estão inseridos quer por operações autónomas, pelo que, para não existir falhas num nó, este tem que conseguir ter um autodiagnostico, autocalibração e autorrecuperação;
- Autonomia – O tempo de vida tem de ser bastante razoável, pois existem muitos cenários em que é importante que a bateria dure um certo tempo, para evitar que o utilizador troque as baterias dos seus dispositivos constantemente. Isto significa que os dispositivos têm que ter uma limitação de potência, ou uma fonte de energia externa, como por exemplo células fotovoltaicas, de modo a carregar as baterias para uma maior duração;
- Baixo custo – Os componentes utilizados têm que ser significativamente baratos, pois como se trata de uma rede, implica a compra de mais do que um dispositivo;
- Tamanho reduzido – os nós podem ser utilizados em diferentes tipos ambientes, sendo que por vezes é melhor que o tamanho do dispositivo seja pequeno impondo também uma redução no consumo;
- Autoconfiguração – Os sensores têm que ser capazes de se autoconfigurarem em caso de falhas, quer dos próprios nós quer existam modificações na topologia da rede;
- Segurança – A rede de sensores tem que ter mecanismos de segurança de modo a garantir a privacidade do utilizador e também impedir acessos não autorizados ou ataques por parte de piratas informáticos.
- Fiabilidade – Para garantir que os dados cheguem corretamente ao seu destino e inalterados, a rede tem que ter um controlo de erros e mecanismos de correção, pois a mensagem pode sofrer alterações devido ao ruído, variações de tempo entre canais e propagação de erros.
- Qualidade de Serviço – Dependendo da aplicação pretendida para a rede de sensores, a qualidade de serviço tem de variar, pois o serviço pretendido pode não querer dar importância ao tempo de atraso da mensagem mas sim dar a uma mensagem perdida.

As WSN podem ser bastante utilizadas na Domótica (do latim *Domus* (Casa) + Robótica), que tem como conceito a utilização da robótica, de forma a trazer a maior segurança e conforto nas habitações. A sua utilização pode implicar a existência de sensores para controlo automatizado.

Os sensores podem ser utilizados também para fins comunitários. No caso concreto da empresa IrRADIARE, a existência de uma rede de sensores que possam controlar a qualidade do ar, pode tornar-se muito menos dispendioso do que instalar uma central de controlo, por exemplo, no centro do Terreiro do Paço. [23]

2.2.1 Arquitetura das WSN

A maior parte dos dispositivos WSN têm uma arquitetura que segue o modelo OSI. Este modelo é composto, como apresentado na figura 2.5, por sete camadas: Física, *Data Link*, Rede, Transporte, Sessão, Apresentação e Aplicação. [24]

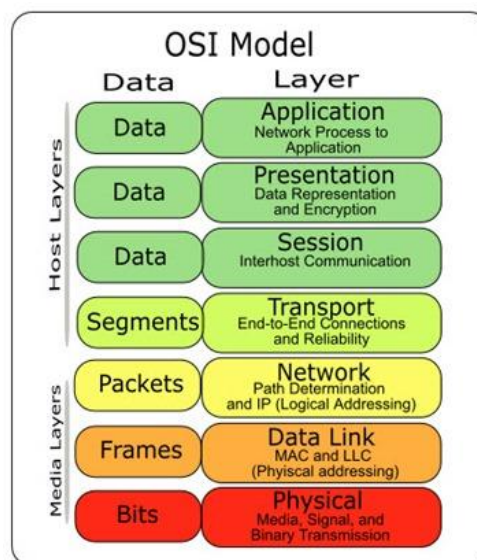


Figura 2.5 - Arquitetura de acordo com o Modelo OSI [25]

Camada Física

A camada física é a camada de mais baixo nível do modelo e tem como principal objetivo transmitir ou receber um conjunto de bits para a camada *Data Link*. Transforma os impulsos elétricos em bits, ou vice-versa, isto é, lida com interfaces mecânicas ou elétricas e com o meio de transmissão. Esta camada é também responsável por dizer se a transmissão pode ser realizada nos dois sentidos simultaneamente ou não.

Camada *Data Link*

Nesta camada os pacotes de dados são codificados e decodificados em bits. Esta deteta e corrige erros que possam acontecer na camada física. Existe uma subcamada denominada MAC (*Media Access Control*) que vai determinar quando é que o nó tem direito/permissão para aceder ou transmitir. Controla também o tráfego de mensagens de forma a impedir o congestionamento, informando assim o recetor quando existe espaço no *buffer* para receber informação.

Camada de Rede

É definido como é que os pacotes de mensagens vão ser encaminhados até ao seu destino. Estes podem ter um caminho estático/pré-definido ou um caminho dinâmico. O controlo de fluxo de mensagens é feito nesta camada, pelo que fica responsável da qualidade de serviço, ou seja, é responsável pelo tempo de resposta, instabilidade, tempo de envio, etc..

Camada de Transporte

É responsável também pela garantia da transmissão de dados de um nó para outro através de um mecanismo de controlo de erro adaptado de acordo com a fiabilidade induzida pela camada de aplicação. Esta camada assegura que toda a mensagem chegará corretamente ao seu destino, podendo essa mensagem ser dividida em fragmentos caso necessário. O tipo de serviço pode ser alterado quando a conexão é estabelecida, por exemplo, pode não ser pretendido que as mensagens cheguem na ordem em que foram enviadas.

Camada de Sessão

Como o nome indica, a camada permite que diferentes dispositivos estabeleçam sessões entre eles, mantendo uma sincronização de modo a garantir que, se existir uma falha, a transmissão seja retomada do ponto onde ocorreu a falha.

Camada de Apresentação

A seguinte camada está vocacionada para a sintaxe e semântica das informações transmitidas, tornando possível a comunicação entre dispositivos com diferentes representações de dados.

Camada de Aplicação

A camada de Aplicação é a camada que está mais relacionada com o utilizador final, por exemplo, no caso de um utilizador procurar uma página de internet, é enviado um pedido ao servidor, utilizando o protocolo HTTP, que posteriormente apresenta a página desejada ao utilizador.

2.2.2 Arquitetura ZigBee e Sinapse

As principais escolhas recaem sobre o ZigBee e o Synapse. Ambos seguem o modelo OSI apresentado acima, com umas pequenas modificações de acordo com os protocolos criados pelos dois (estes têm a arquitetura apresentada na Figura 2.6).

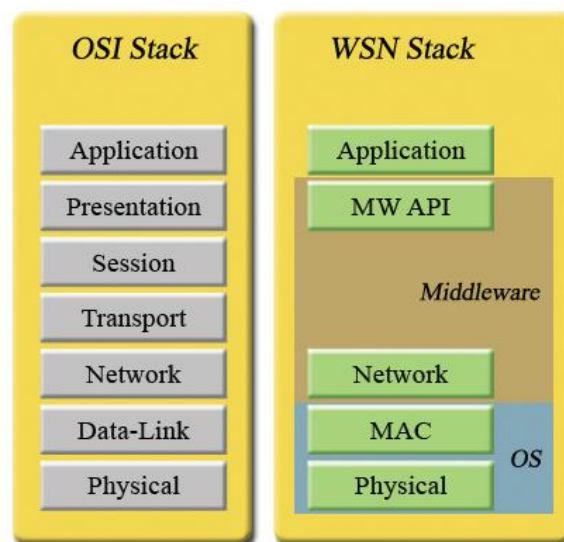


Figura 2.6 - Semelhanças do modelo OSI utilizadas no ZigBee [26]

No caso do ZigBee a camada física e MAC estão definidas de acordo com o protocolo IEEE 802.15.4, que trata das operações de modulação e desmodulação dos sinais transmitidos e recebidos, respetivamente. A tabela 2.1 demonstra os valores típicos da camada física.

Tabela 2.1 – Frequências típicas da camada física [27]

	Espaço	Velocidade	Num. De Canais
2.4GHz	Universal	250kbps	11-26
868MHz	Europa	20kbps	6
915MHz	América	40kbps	1-10

A camada de MAC é responsável pela transmissão de dados acedendo a diferentes endereços com CSMA (*Carrier Sense Multiple Access Collision Avoidance*).

A camada de Rede é descrita pelos fabricantes dos dispositivos que normalmente é denominada por camada ZigBee, onde são também apresentados os mecanismos de encriptação. Esta é responsável pelo encaminhamento, configurações dos dispositivos e conexão dos dispositivos finais.

A camada de aplicação está dividida em três partes: subcamada de aplicação (APS), aplicação Framework (AF) e dispositivos finais. A camada contém a aplicação que corre na camada de rede, sendo que um nó pode conter mais do que uma aplicação, como por exemplo, uma para medir temperatura, outra para medir a pressão e outra para medir a humidade. A subcamada APS pode disparar instruções necessárias para a aplicação específica, de modo a que seja executada uma ação, como por exemplo, ligar um LED. [20] [28] [29]

Por outro lado, a arquitetura do SNAP (Synapse Network Application Protocol) foi criada de modo a que o utilizador, mesmo que não tenha o mínimo de conhecimento acerca de redes sem fios possa, mesmo assim, utilizá-la e criar aplicações que executem nos nós pretendidos. Uma visão de mais alto nível da arquitetura do SNAP é representada na figura 2.7, que foi desenhada de forma a correr com a máxima eficiência num microcontrolador de 8bits.

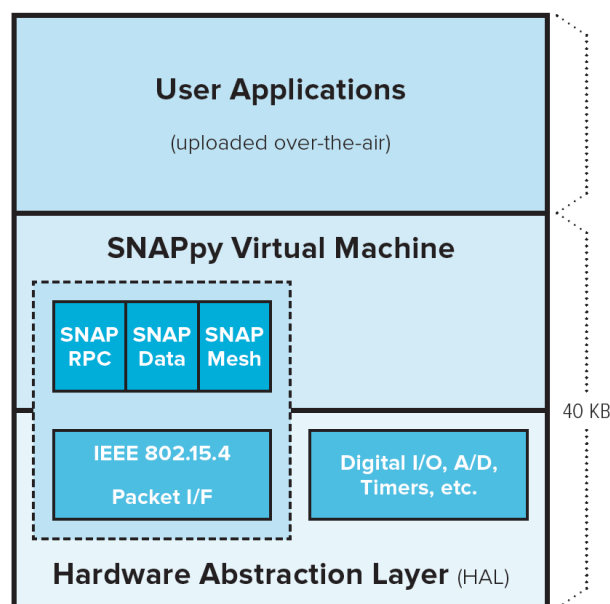


Figura 2.7 - Arquitetura Synapse [19]

A camada física e a camada MAC, tal como anteriormente explicado no ZigBee, também elas estão definidas através do protocolo IEEE 802.15.4. As restantes camadas permitem que o utilizador final possa desenvolver aplicações que vão ser compiladas na Máquina Virtual SNAPpy (*SNAPpy Virtual*

Machine). No caso da camada de Aplicação, é o utilizador que cria a sua própria aplicação. A subcamada SNAP RPC (*Remote Procedure Call*) é bastante importante pois, permite que o utilizador invoque funções presentes noutros nós, através da rede em que estão ligados.

Como o Synapse é um produto em desenvolvimento (apesar de já possuir alguns produtos para comercialização) ainda não existe muita informação do mesmo, ficando sem conhecimento do que realmente se processam as restantes camadas.

2.2.3 Encaminhamento de mensagens

Nas WSN o tipo de encaminhamento de mensagens é um pouco diferente dos encaminhamentos convencionais em redes fixas. Existem muitos fatores que são mais importantes para as WSN do que para as redes convencionais, como por exemplo, a inexistência de uma infraestrutura, os nós dos sensores podem falhar, a rede tem que ser escalável, heterogénea e existe ainda muitos requisitos em relação à utilização de energia.

Existem alguns protocolos para o encaminhamento de mensagens definidos para este tipo de redes, podendo ser divididos em protocolos: baseados na localização (Location-based Protocols), baseados em hierarquia (Hierarchical-based Protocols), baseados na mobilidade (Mobility-based Protocols) e oportunistas. [30] [31] [32]

No ZigBee, o encaminhamento de mensagens utiliza um algoritmo *Cluster-Tree* ou uma simplificação do algoritmo AODV (*Ad hoc On-Demand Distance Vector*) – o algoritmo AODVjr (figura 2.8).

A simplificação do algoritmo AODV tem em conta o baixo custo e consumo. Este não utiliza um número de série do nó de destino, mas estimula a localização do nó de destino através de um único pacote que este é capaz de responder, ou seja, mesmo que passe por outros nós, somente o nó de destino é que irá responder.

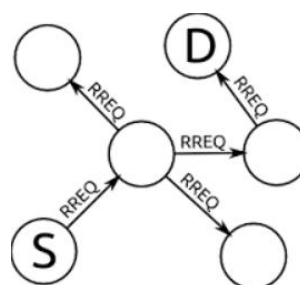


Figura 2.8 - Algoritmo AODVjr [33]

No Synapse, no encaminhamento de mensagens, é utilizado um protocolo oportunista. Este não utiliza nenhum protocolo de encaminhamento de mensagens específico, apenas invoca funções RPC usando o endereço de cada nó da rede. O sistema de encaminhamento de mensagens adotado SNAP foi um sistema ad-hoc baseado no *Standard* DYMO (Dynamic Manet On-demand). O protocolo DYMO tem como principais operações a descoberta e gestão. A operação de descoberta, como o nome indica, serve para detetar qual o nó destino, que quando é reconhecido, envia uma mensagem de volta pelo mesmo percurso. Quando o nó que fez o pedido receber essa resposta, estabelece a ligação entre eles [34]. Este protocolo é bastante idêntico ao algoritmo AODV usado no ZigBee.

Os caminhos são pré estabelecidos assim que necessário e, o estado da rede, é constantemente atualizado através de tabelas que especificam quando existe alterações na rede. O SNAP, de forma a lidar com efeitos dinâmicos dos RF, suporta Link Quality (figura 2.9) baseados em parâmetros de encaminhamento, isto é, o SNAP identifica todos os nós presentes na rede com diferentes pesos relativos, sendo esse peso determinado pelo canal, largura de banda e intensidade do sinal. De acordo com o resultado, é escolhido o caminho a percorrer para que a informação chegue corretamente ao seu destino. [35] [36]

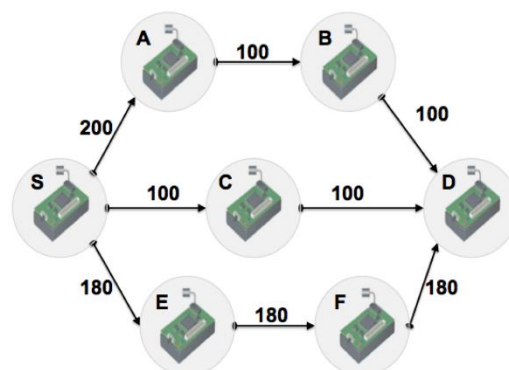


Figura 2.9 - Algoritmo utilizado pelo Synapse [37]

A principal função do encaminhamento de mensagens no SNAP é a capacidade de gestão das rotas armazenadas na RAM de cada nó, aumentando assim a velocidade da formação das novas rotas e minimizando o número de pedidos a todos os nós para saber qual o que tem menos peso. No caso do módulo RF200 suporta até 100 entradas, para armazenamento das rotas.

2.2.4 Segurança

A segurança é um tema bastante importante quando se faz um projeto de redes sem fios, pois os utilizadores não querem os seus dados expostos e disponíveis para todos, nem querem receber dados que não sejam fidedignos, com a possibilidade de piratas informáticos os interceptarem e alterá-los. Desde 1990 que os algoritmos de segurança têm sofrido bastantes atualizações.

Os tipos de segurança mais conhecidos são utilizados pela tecnologia Wi-Fi, mais concretamente o WEP (*Wired Equivalent Privacy*), WPA (*Wi-Fi Protected Access*), WPA2 (*Wi-Fi Protected Access II*). No WEP a segurança não era propriamente forte devido a certas restrições que naquele tempo existiam, pois só permitiam que o tamanho da chave fosse até 64bits. Após essas restrições, passou a ser possível uma chave até 256bits. Mas, posteriormente veio a ser publicado a facilidade que era descriptar uma chave feita com este algoritmo.

O WPA2, como se pode verificar é uma evolução do WPA pois, tal como o WEP, o WPA apesar de utilizar um algoritmo de encriptação TKIP (*Temporal Key Integrity Protocol*), também a sua segurança foi comprometida. Já o WPA2 veio melhorar a segurança dos utilizadores e dificultar os piratas informáticos, devido ao seu algoritmo de encriptação AES (*Advanced Encryption Standard*). Mas ainda assim é possível derrubar esta encriptação, sendo necessário que os piratas informáticos gastem cerca de duas a catorze horas e que tenham um computador capaz de processar bastante informação. [38]

Nesta dissertação existem dois tipos de segurança, a segurança na privacidade do utilizador e a segurança na integridade das mensagens. Sendo que a melhor forma de garantir toda a segurança, passa principalmente pela encriptação das mensagens.

Existem vários mecanismos de encriptação, como por exemplo, SHA (Secure Hash Algorithm), MD5 (Message-Digest algorithm 5), AES, TKIP, entre outros.

O SHA e o MD5 usam funções Hash. Estas funções são também conhecidas como encriptações de um único sentido, sendo que um valor único de Hash, de comprimento fixo, é calculado com base no texto que se quer encriptar, o que torna impossível (até à data) reverter a situação. Esta encriptação funciona como uma impressão digital única e têm um uso muito comum na encriptação de palavras passe. Nos casos específicos, o MD5 foi desenvolvido pelo Ron Rivest que gera, independentemente do tamanho da mensagem a encriptar, valores Hash de 128bits. O SHA, também desenvolvido pelo Ron Rivest, é bastante idêntico ao MD5 sendo um pouco mais forte e possui diferentes tamanhos de valores Hash pelo que pode ser SHA-1, SHA-224, SHA-256, SHA-384

e SHA-512, tendo um tamanho de 160, 224, 256, 384 e 512 bits respetivamente. Este tipo de encriptação funciona para garantir uma autenticação segura para o utilizador que, quando registado, é gerada uma chave Hash única para cada um dos dados enviados, ou seja, uma para o nome de utilizador e outra para a palavra passe. [39] Supondo uma palavra passe como “David”, a chave de encriptação gerada é:

```
0f4e49e88d2b8b8501a4ba7a2a320a1e496c2d4a1d1cba1c068822cf690a339349fdc3d3d8545f972  
36e3f07f9cf40eefd8f590ca3f8aea4919adef66361b506
```

O AES é composto por três blocos de cifras, o AES-128, AES-192 e AES-256, que cada um encripta e descripta as mensagens em blocos, usando chaves com tamanho de 128, 192 e 256 bits, respetivamente, necessitando também de uma grande capacidade de processamento. [40] O protocolo 802.15.4, utiliza um algoritmo de encriptação denominado AES (*Advanced Encryption Standard*) com uma chave de encriptação de 128 bits o que perfaz uma complexidade computacional de $2^{126.1}$. Assim os dados enviados entre os dispositivos, ou seja, os dados recolhidos pelos sensores e enviados pela rede (sem fios) até chegar ao dispositivo principal, não ficam facilmente acessíveis a qualquer pessoa que os tente intercetar.

No caso da página de internet, a comunicação feita entre servidor e a própria página, que é executada por serviços Web (são serviços que respondem a perguntas previamente definidas), está protegida com um método de encriptação denominado SSL (Secure Sockets Layer). Normalmente, sem este tipo de proteção os dados enviados entre clientes e serviços Web são vulneráveis, pois são enviados como texto dito “normal”. Com o protocolo de segurança SSL, o algoritmo encripta todo o tipo de informação que passe entre eles. [41]

2.2.5 Recolha de Energia (*Energy Harvesting*)

No que toca à rede de sensores sem fios, é importante que o consumo de energia seja bastante reduzido, de modo a maximizar o tempo de vida das baterias. Os dispositivos que foram dimensionados para trabalhar numa WSN consomem muito pouca energia, dependendo do fabricante e do tipo de processamento que tenha que executar.

Supondo que a bateria duraria um ano, sem ser necessária a intervenção humana para a carregar, se fossem utilizadas fontes externas para carregar a bateria, logicamente duraria muitos anos. Isto depende do tipo e de quantas fontes de energia externa seriam necessárias para manter o dispositivo a trabalhar durante anos sem ser necessário nenhuma intervenção.

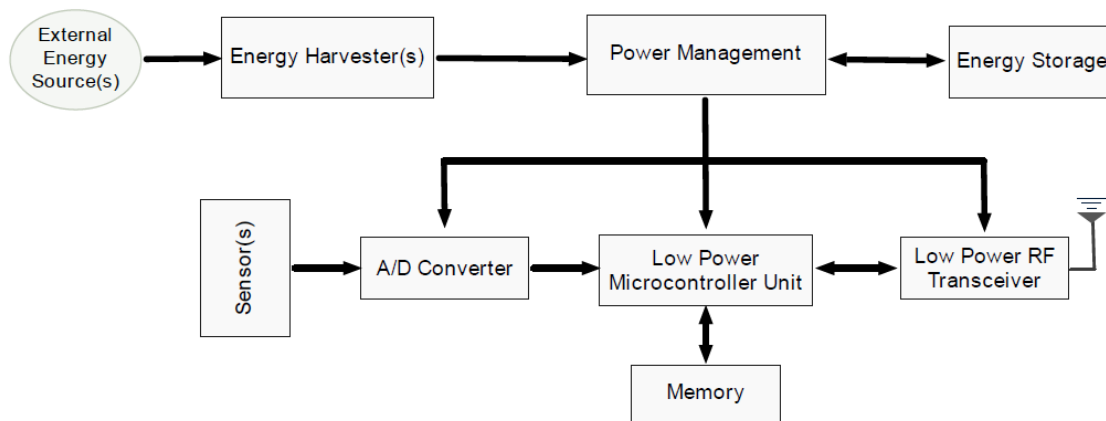


Figura 2.10 - Modelo comum de Recolha de Energia [42]

Existem muitos modelos de recolha de energia, seguindo sempre o modelo representado na figura 2.10. Um exemplo de um dos modelos em estudo está apresentada na figura 2.11 que como se pode observar, a energia pode ser usada diretamente para o sensor (caso necessite de energia para funcionar). Este tem incluindo também uma bateria, de forma que, se a recolha de energia (proveniente de uma célula pucovoltaica, por exemplo) não for suficiente para alimentar o sensor, será retirado energia da bateria. Mas quando a energia recolhida é em demasia para o funcionamento do sensor, existe um controlo de energia, em que certa parte irá para o sensor e a restante será armazenada na bateria. [43] [42]

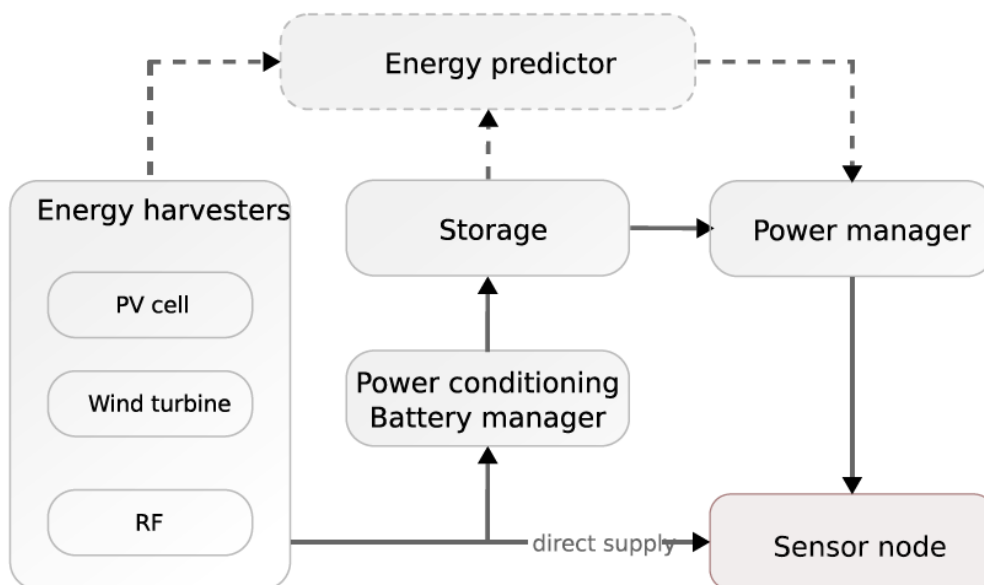


Figura 2.11 - Modelo de Recolha de Energia [42]

2.3 Sensores

“Tal como acontece com muitas tecnologias, as aplicações de defesa têm sido um motor de pesquisa e desenvolvimento nas redes de sensores.” [44]. Com o passar dos anos os sensores têm evoluído cada vez mais e foram abrangendo as mais diversas áreas de investigação de modo a facilitar e a controlar tudo à sua volta. Como pode ser verificado na figura 2.12, demonstra as suas aplicações de acordo com o passado e o presente, através da procura de mercado, relacionado com a energia consumida por sensor.

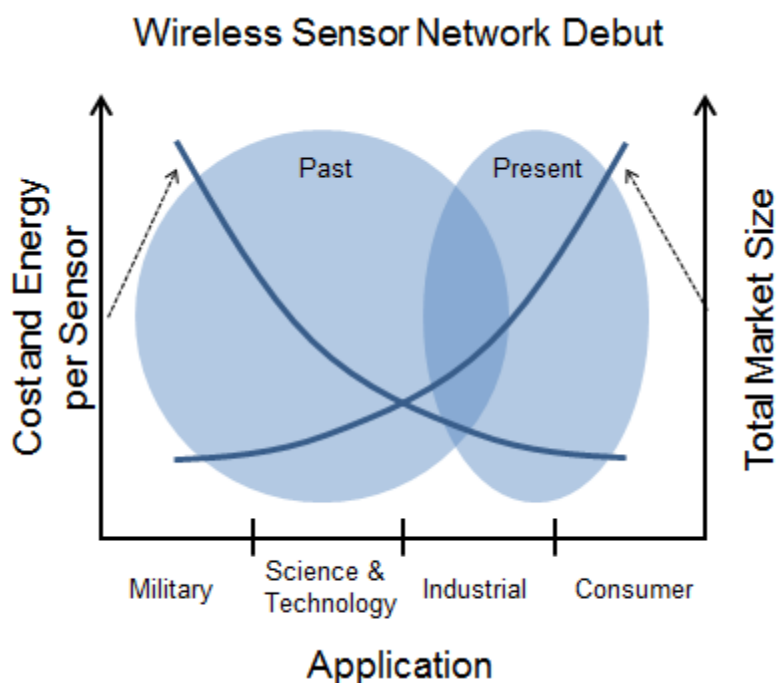


Figura 2.12 - Evolução na utilização da Rede de Sensores [45]

Os sensores são uma mais-valia para a sociedade e com a evolução da Internet das Coisas, mais do que nunca, a sua utilização tem vindo a expandir-se, como por exemplo, na segurança das casas, meio militar. Neste capítulo serão mencionados alguns sensores existentes, principalmente na perspetiva da empresa IrRADIARE.

2.3.1 Tipos de sensores

Existem muitas aplicações para a utilização dos sensores e, para tal, tem que existir diferentes tipos de sensores para cada uma das aplicações desejadas, pelo que para entender as WSN é necessário compreender um pouco do funcionamento dos sensores. [45]

Os sensores são dispositivos que convertem um parâmetro físico como temperatura, fluxo sanguíneo, humidade, pressão, etc., num sinal elétrico de modo a ter essa informação em formato digital. Os sensores podem ser escolhidos de acordo com os seguintes critérios [46]:

- Precisão – os dados recolhidos têm que ter um mínimo de precisão para não ocorrer erros de leitura;
- Condições atmosféricas – é necessário ter em conta qual o tipo de ambiente a que os sensores vão ficar expostos, pois existem limites mínimos e máximos para os quais estão dimensionados;
- Alcance – as medições feitas pelo sensor podem conter um limite mínimo ou máximo, dependendo do tipo de sensor.;
- Custo – o critério qualidade/preço tem que ser bem estudado;
- Calibração – as leituras podem sofrer alterações ao longo do tempo, quer seja por desgaste, quer seja pelas condições a que estava exposto.

A maior parte dos critérios acima descritos podem ser confirmados no *Datasheet* de cada sensor, ato a ser executado após a escolha do tipo de sensor.

Sensor de temperatura

O sensor de temperatura é sem dúvida um dos mais utilizados em qualquer área. Está presente em praticamente todos os dispositivos, podendo ser mecânico ou eletrónico, requerendo ou não contato físico direto com o dispositivo. O comportamento do sensor de temperatura é equivalente ao termómetro de mercúrio, em que o mercúrio expande e contrai consoante a temperatura. Entre eles estão termostato e termostato bi-metálico, termístor, RTD (*Resistive Temperature Detector*), termopar (*Thermocouple*), como apresentado na figura 2.13.

Os termopares são bastante eficazes quando se quer medir grandes diferenças de temperaturas, como por exemplo de zero a quinhentos graus centígrados. Estes têm um tempo de resposta significativamente baixo e, a sua precisão, pode ser inferior a um grau, dependendo do fabricante. Os termopares são, basicamente, dois fios (exatamente do mesmo material) que a extremidade fusão está exposta, de forma a medir a temperatura. As pontas são então ligadas a terminais que através da corrente que passa pelo fio, gera uma diferença de potencial, cujos valores podem ser consultados no *Datasheet* do material. Como é o caso de um termopar do tipo-k 3 que, por

exemplo, se a temperatura for de 20 graus centígrados então a diferença de potencial é igual a 0.919mV. [47]

Os RTD são relativamente mais caros que os termopares. Estes são utilizados para medições exatas da temperatura com um erro de ± 0.019 graus centígrados e possuem um menor alcance.

Os termístores são praticamente iguais aos RTD, sendo que a sua principal diferença é que requerem uma tensão de excitação ao invés de corrente de excitação necessária para os RTD. O seu erro absoluto é de ± 0.05 graus centígrados e são mais baratos que os RTD.



Figura 2.13 - Tipos de Sensores (Termopar, RTD, Termístores respetivamente) [48] [49] [50]

Sensor UV (Ultra Violeta)

Os sensores Ultra Violeta (figura 2.14), medem a intensidade ou potência incidente de raios Ultra Violeta no sensor. Estes sensores servem, por exemplo, averiguar se a exposição ao sol (numa praia) num certo período horário é, ou não, prejudicial para a pele humana, pois esta é uma radiação que tem uma largura de banda que não é visível ao olho humano. O sensor UV tem também diversas aplicações em automóveis, farmácia, robótica, entre outros. É usado um diamante policristalino, que a luz incidente nele é refletida para um detetor que vai detetar a intensidade da radiação UV presente. [51]



Figura 2.14 - Sensor UV [52]

Sensor de Monóxido de Carbono

Ao contrário do que se pensa, o dióxido de carbono não é o gás mais preocupante quando se trata da saúde da população em geral. O monóxido de carbono é um gás incolor e inodoro que pode matar repentinamente. Normalmente estes sensores (figura 2.15) são usados em parques subterrâneos, túneis, oficinas, etc. O princípio de funcionamento é igual ao dos sensores de temperatura, mas este tem um semicondutor de metal óxido, que altera a resistência interna de acordo com a quantidade de monóxido de carbono existente. Essa alteração faz disparar um relé que normalmente ativa as ventoinhas das condutas de ar. [53]



Figura 2.15 - Sensor de Monóxido de Carbono [54]

Existem muitos outros sensores, que não serão falados pormenorizadamente, mas todos têm praticamente o mesmo princípio, que passa por alterar a corrente elétrica (através da resistência interna dos sensores) criando uma queda de potencial aos seus terminais, que posteriormente com os valores fornecidos pelo *Datasheet* do fornecedor, pode-se obter um valor conhecido aos olhos do utilizador. Alguns dos outros sensores mais conhecidos são mencionados em baixo:

- Sensores de som
- Sensores de vibração;

- Sensores de deformação;
- Sensores de movimento;
- Sensores de luminosidade;
- Sensores de pressão;
- Sensores de humidade;
- Sensores de fumo;
- Sensores de presença;
- Sensores de proximidade;
- Sensores indutivos;
- Sensores capacitivos.

As seguintes secções estão apresentadas de forma a contextualizar todo o projeto, pois serão utilizados termos que estão descritos mais pormenorizadamente na dissertação Ponto de Acesso Móvel em Ambiente Sensorial, realizado por José Ricardo de Lima Abrantes.

2.4 Plataformas para telemóveis

Os sensores, após recolherem os dados necessitam de os enviar para a *Cloud*, permitindo que o utilizador possa aceder a essa informação. Para tal, é utilizado o telemóvel como meio intermediário para recolher os dados através do Bluetooth (enviados pelo Bluetooth anexado ao Arduino) e posteriormente quando tiver acesso à internet poderá assim enviá-los para a *Cloud*.

Existem várias plataformas para o telemóvel, estando entre elas a plataforma Android, iOS, Windows Phone, Blackberry, entre muitos outros. Optou-se por desenvolver uma aplicação para a plataforma Android visto que é a plataforma mais utilizada (Tabela 2.2 [55] [56]).

Tabela 2.2 - Comparação da utilização de sistemas operativos nos Smartphones

Período	Android	iOS	Windows Phone	BlackBerry OS	Outros
2015	78.0%	18.3%	2.7%	0.3%	0.7%
2014	81.2%	15.2%	2.5%	0.5%	0.7%
2013	75.5%	16.9%	3.2%	2.9%	1.5%
2012	59.2%	22.9%	2.0%	6.3%	9.5%

Como surgiu e o que é a plataforma Android?

Android começou por ser uma empresa única em 2003, fundada pela Andy Rubin, com o objetivo de criar um sistema operativo avançado para câmaras digitais. Acabou por desistir da ideia e implementar o sistema operativo em telemóveis. Em 2005 foi comprada pela Google e em 2008, com parceria da T-Mobile, lançou o primeiro telemóvel com a plataforma Android. [57]

O nome de cada versão do Android foi dada depois de uma sobremesa, pelo que já conta com a versão Cupcake, sucedida pela Donut, Eclair, Froyo (gelado de logurte), Gingerbread, Honeycomb, Ice Cream Sandwich, Jelly Bean, Kitkat e por fim as versões atuais Lollipop e Marshmallow. [58]

A plataforma Android é um *software* aberto e gratuito que usa um sistema modificado em Linux Kernel, sendo as suas aplicações (grande parte delas) são desenvolvidas em Java. Por ser um *software* aberto, proporciona ao utilizador a possibilidade de modificar o código alterando, por exemplo, a frequência do processador (*CPU Clock*), permitindo assim que a bateria dure mais, caso se reduza a frequência ou no caso de querer melhorar o desempenho do mesmo, é aumentando a frequência. Para desenvolver aplicações Android é necessário a instalação de um *software* denominado SDK (*Software Development Kit*) que é baseado em Java.

A aplicação desenvolvida foi baseada através de um *Open Source* que permite a comunicação com o Bluetooth. Essa aplicação foi implementada para realização de simples testes como o envio e receção de dados pelo Bluetooth.

2.5 Serviços Web e Cloud

Os dados recebidos pelos sensores são enviados para o telemóvel que, posteriormente, quando o utilizador tiver acesso à internet, esses dados são enviados para a *Cloud*. Parece ser direto, mas não o é.

O conceito *Cloud* (figura 2.16) começou a ganhar uma maior importância a partir de 2006 quando a empresa Amazon lançou o serviço *Amazon Elastic Computing Cloud*. A *Cloud* disponibiliza serviços que possibilitam, por exemplo, o armazenamento de dados em servidores na internet, que estão colocados em *Data-Centers*. Um utilizador normal, de um computador ou telemóvel, está habituado a guardar os seus documentos (fotografias, filmes, documentos de texto, etc.) no armazenamento interno dos dispositivos. A *Cloud* veio melhorar essa experiência, pois o utilizador pode agora guardar todos os seus documentos na *Cloud*, evitando que sejam perdidos, por falha do dispositivo ou que o disco/circuito de armazenamento tenha sido corrompido. [59]

Existem algumas vantagens na sua utilização como, por exemplo, o aumento dinâmico dos recursos necessários e os dados guardados podem ser facilmente acedidos em qualquer lado, desde que exista uma ligação à internet. Também existem desvantagens como, por exemplo, caso não exista uma ligação à internet, e os documentos não tenham sido descarregados para o computador previamente, o utilizador não vai conseguir ter acesso aos mesmos, a segurança e privacidade dos documentos por vezes pode não ser a melhor e ainda não existe um modelo *standard* de forma que os utilizadores possam passar os seus documentos de uma *Cloud* para outra.



Figura 2.16 - Conceito *Cloud* [60]

A *Cloud* é importante neste projeto pois, é através dela que o utilizador vai conseguir ter acesso à informação recolhida pelos sensores, podendo assim acede-la em qualquer lugar, desde que tenha uma ligação à internet. Também tem a vantagem de o utilizador não sobrecarregar a capacidade do telemóvel e do cartão SD colocado no nó principal. Quando o telemóvel se ligar à internet, os dados são automaticamente enviados para a *Cloud* e, posteriormente, será comunicado ao nó principal que os dados foram corretamente recebidos, podendo assim apagar esses dados e libertar espaço para novos dados no cartão SD.

2.6 Projetos Similares

Como é normal, existem bastantes projetos que envolvem a utilização de rede de sensores e que mantém o utilizador sempre atualizado dos mesmos sempre que desejar.

Em primeiro lugar, para se perceber as diferenças para os restantes projetos já existentes no mercado, é necessário referir que o projeto/conceito desenvolvido em parceria com a empresa IrRADIARE tem a particularidade de não depender de nenhum ponto fixo. Ou seja, os dispositivos podem ser colocados em qualquer lugar, sendo a sua única limitação o alcance do Bluetooth e o número de pessoas que se aproximem dos mesmos.

Valarm é uma empresa que proporciona o utilizador a ter toda a informação recolhida pelos sensores, disponível no telemóvel e na internet. Os sensores são todos ligados por um cabo USB OTG que têm, por sua vez, uma alimentação por USB que está ligada a um telemóvel. A grande vantagem desta tecnologia é a utilização do cabo USB OTG e da sua grande aplicação que permite que sejam ligados os mais diversos sensores ao telemóvel. O problema será no caso do utilizador querer instalar sensores em vários pontos distantes, cada “posto” terá que possuir um telemóvel, como mostra a figura 2.17, o que implica que também tenha que pagar a uma operadora de telefone para poder enviar os dados para a *Cloud* (no caso da inexistência de um ponto de Internet por perto). Para evitar taxas e se for preferível, o próprio utilizador pode descarregar os dados através de outro telemóvel através de Bluetooth [61].



Figura 2.17 - Produto Valarm com a utilização de um telemóvel como *gateway* [61]

O SensorCloud por LORD MicroStraing tem o mesmo conceito presente nesta dissertação, que passa por ter um *gateway* como dispositivo intermediário para receber informação (figura 2.18). Neste caso o *gateway* é um retransmissor (WSDA – 1500 – LXRS) que transmite os dados até 2km. O retransmissor permite acesso pessoal, acesso público e pode ser programado da forma que o utilizador entender. Este recebe informação através de comandos de pedidos http e os dados são posteriormente apresentados num ficheiro Excel. Este implica que, mais uma vez, exista um ponto de internet por perto para que o retransmissor possa enviar os dados para a *Cloud*. Os dados só serão enviados quando o utilizador assim o entender, pois é necessário que este aceda ao retransmissor através de um página http e carregar num botão para serem enviados os ficheiros. [62]



Figura 2.18 - SensorCloud com um *gateway* (necessário ligação à internet) [62]

Um último exemplo é o caso do Libelium. Este também usa um *gateway* como intermediário, só que neste caso é um computador portátil ou fixo. A particularidade destes dispositivos é que possuem a capacidade de comunicar através de IPV6 (figura 2.19), ou melhor, 6LoWPAN (*IPv6 over Low power Wireless Personal Area Networks*). Isto permite que muitos mais dispositivos possam ser ligados à rede e também, como é uma tecnologia de baixo consumo, as baterias dos dispositivos duram mais tempo. Posteriormente o coordenador converte para IPV4, para enviar os dados recebidos para o portátil. Estes dispositivos são *boards* que são personalizáveis, em que o utilizador pode adicionar novos sensores que passarão a ser reconhecidos pela mesma. [63]

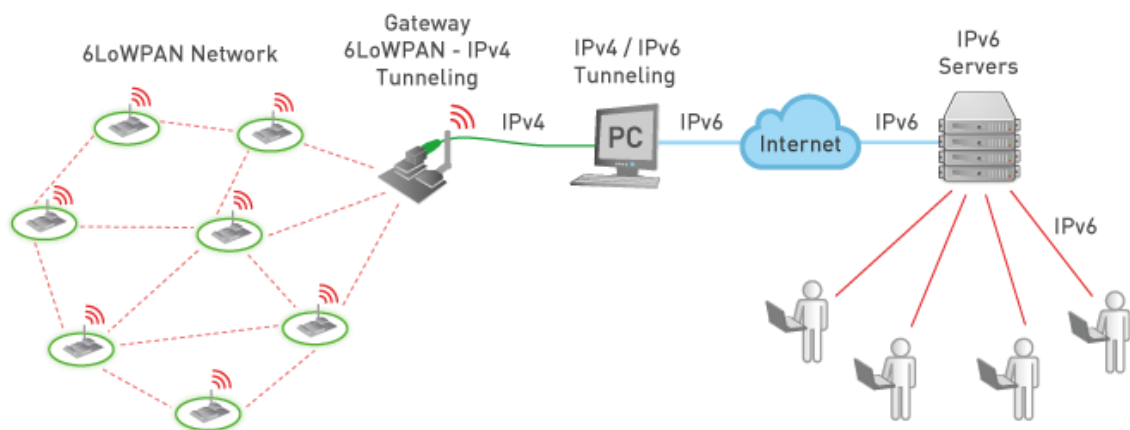


Figura 2.19 - Libelium com um *gateway* ligado por cabo ou sem fios (necessário ligação à internet) [63]

3

Proposta

3.1 Proposta para solução do problema

De acordo com o problema apresentado, é necessário que o utilizador consiga colocar os seus sensores em qualquer lugar, sem depender de um ponto fixo, quer seja para conseguir acesso à internet quer seja para alimentação dos mesmos. Para que tal seja possível é necessário que sensores recolham a informação desejada e útil para o utilizador e que enviem para um dispositivo de forma a processar os dados. Posteriormente esses dados são enviados por uma rede de sensores em Malha.

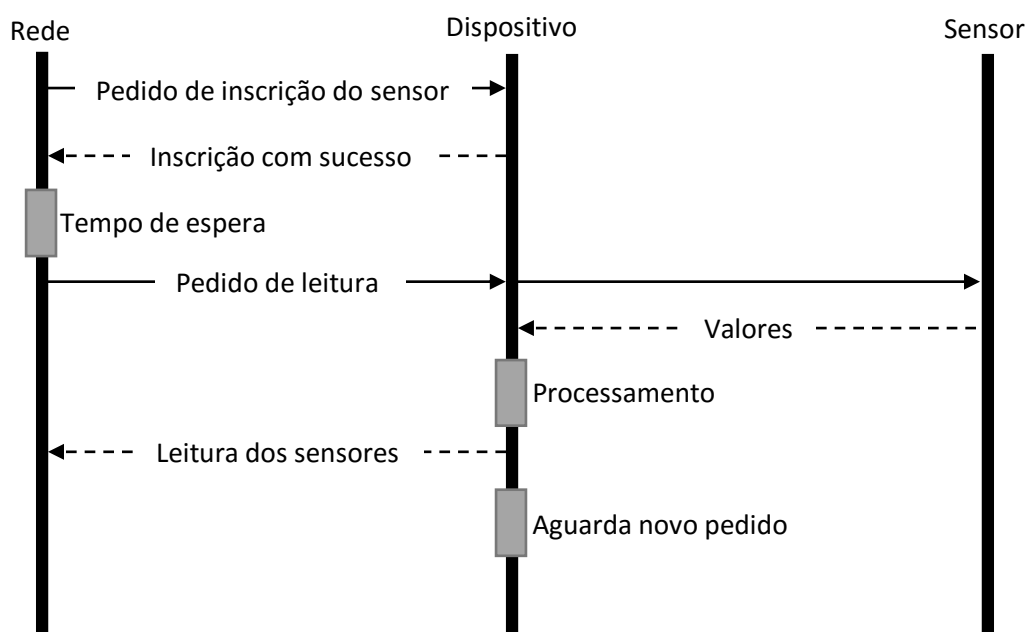


Figura 3.1 – Diagrama de sequência de comunicação entre a rede, dispositivo e sensor

Observando o diagrama de sequência presente na figura 3.1, pode ser verificado que, no caso da comunicação com os sensores, existem três tecnologias associadas: a própria rede em Malha, dispositivos (que constituem essa mesma rede) e os sensores.

Inicialmente, para a integridade de todo o sistema, os sensores têm que ser adicionados aos dispositivos finais (portas *Input/Output*) através de um pedido feito pelo dispositivo principal, pela rede de sensores. Através dessa mesma rede, será enviado um novo pedido (feito pelo dispositivo principal) para leitura dos sensores já registados. O dispositivo final será responsável por todo o processamento de dados dos sensores agrupando toda a informação numa única mensagem, enviado essa mensagem pela rede até ao dispositivo principal. Tanto a rede de sensores, como o próprio dispositivo final têm um tempo de espera que é utilizado para não tornar a recolha de informação tão sistemática.

A rede de sensores terá a topologia em Malha pois o modelo proposto implica que o utilizador possa alterar a disposição, inserir ou retirar dispositivos finais sem pôr em causa a integridade da mesma. O modelo proposto tem como principal objetivo, permitir que o utilizador adicione este projeto em locais menos acessíveis, como por exemplo uma floresta, sendo que a topologia de rede em Malha permite que seja feito um “caminho” (através da floresta) até ao dispositivo principal.

Este foi o modelo proposto para o sistema da rede de sensores, mas sozinha, não resolve o problema referido. A rede de sensores só fará com que a informação passe de dispositivo para dispositivo até chegar a um dispositivo principal.

O dispositivo principal será constituído por, um dispositivo de comunicação, visto que será através dele que toda a informação recolhida pelos sensores será enviada para um *gateway* (telemóvel). Isto permite que utilizador possa colocar o seu sistema de sensores em qualquer lugar, desde que exista pelo menos uma pessoa que se aproxime do dispositivo principal.

O diagrama apresentado na figura 3.2, demonstra toda a comunicação realizada entre o dispositivo principal (que está incluído na rede), dispositivo de comunicação e o *gateway*.

Após a rede ter recolhido a informação de todos os sensores, será feito um pedido de receção dos dados ao dispositivo de comunicação. Este dispositivo é responsável pela comunicação com o *gateway*. Quando o dispositivo comunicação validar o pedido, será executada o envio de dados por parte do dispositivo principal. O armazenamento (servirá para ordenação e salvaguarda de todos os dados recolhidos, pois o *gateway* poderá não estar sempre disponível) e processamento é responsabilidade do dispositivo de comunicação. Assim que todos os dados forem recebidos a rede de sensores será avisada.

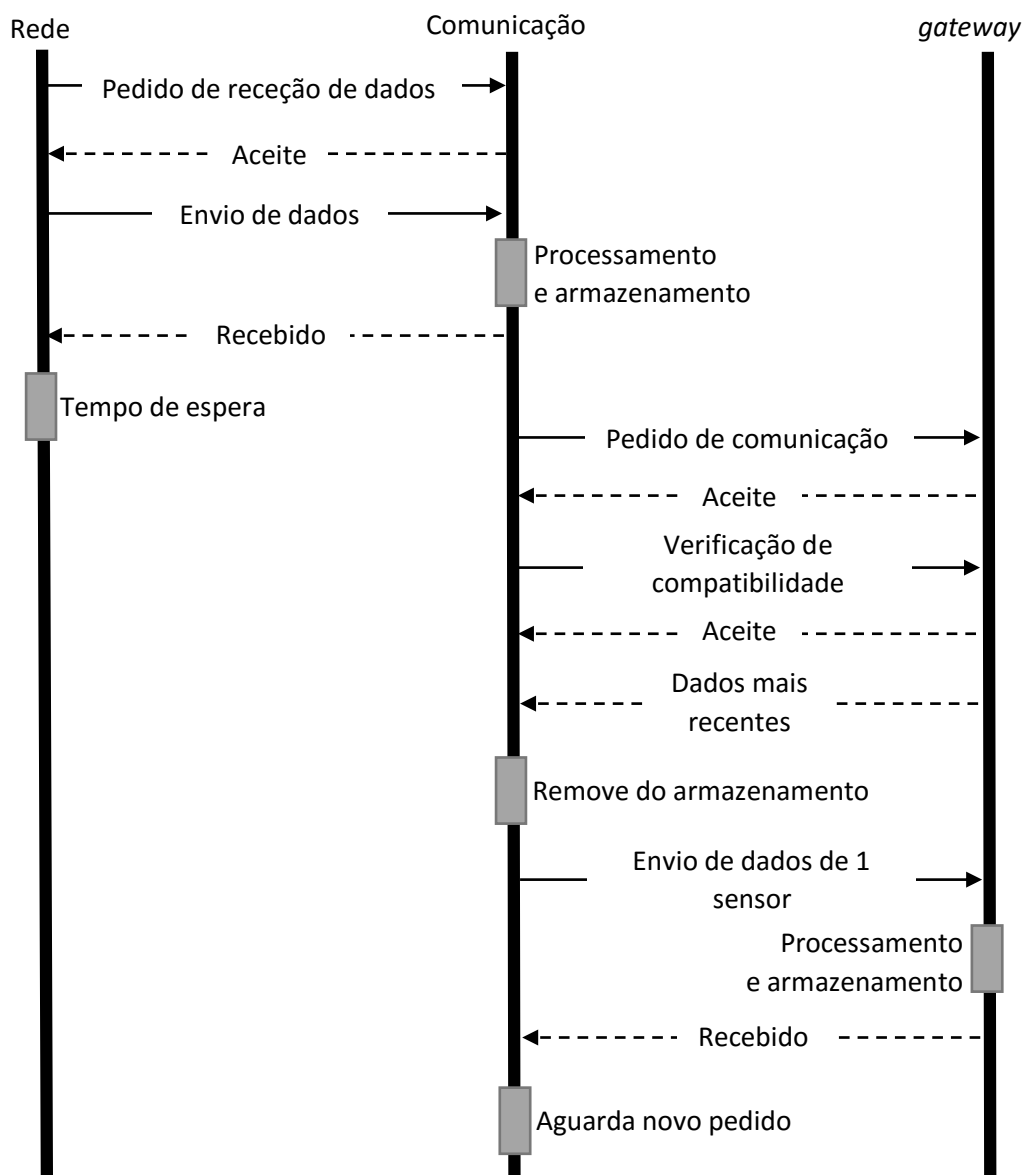


Figura 3.2 – Diagrama de sequência de comunicação entre a rede, dispositivo de comunicação e *gateway*

De seguida, caso o *gateway* esteja disponível, será enviado um pedido para emparelhamento por parte do dispositivo de comunicação. Caso afirmativo, será verificado se o *gateway* cumpre todos os requisitos para recepção dos dados dos sensores. Esta verificação permitirá que o portador do *gateway* não aceda aos dados sem permissões (embora toda a informação não fique diretamente acessível no *gateway*). De forma que os requisitos sejam cumpridos, é necessário que o *gateway* tenha instalado uma aplicação desenvolvida neste projeto. Esta aplicação poderá ser instalada na maior parte dos dispositivos móveis (*gateways*), dos quais contribuíram para que o utilizador da rede de sensores possa ter acesso a toda a informação recolhida pelos sensores.

Se os requisitos anteriores estiverem disponíveis, o dispositivo principal iniciará o envio de dados de um sensor específico, armazenado previamente. A cada ligação estabelecida com o *gateway*, irá

ser confirmado quais os dados que já se encontram disponíveis para o utilizador (na *Cloud*), permitindo assim, que o nó principal possa apagar dados mais antigos. Esta operação irá prevenir que o armazenamento do dispositivo principal fique completamente cheio, garantindo que não será necessário intervenção sistemática do utilizador para recolha de informação dos sensores.

A confirmação dos dados recebidos na *Cloud* será de pelo menos uma vez, ou seja, os dados presentes no armazenamento do dispositivo principal, dependerão apenas de uma única confirmação por parte do *gateway*, que por consequência, os dados serão apagados do armazenamento do dispositivo principal.

Quando o envio de dados para o *gateway* for finalizado, o dispositivo principal ficará então a aguardar novo pedido da rede de sensores ou, caso outro *gateway* esteja disponível, estabelece novamente outra ligação. Os dados recebidos estarão contidos num ficheiro, que pertence a um único sensor, incluindo a data em que foi recolhida essa informação, a própria informação, o nome do proprietário da rede de sensores, a data em foi enviado o ficheiro para o *gateway*. Este ficheiro não terá um tamanho suficientemente grande que prejudique o armazenamento interno do *gateway*.

Quando o *gateway*, tiver acesso à internet, o ficheiro com os dados recolhidos por um sensor específico, será enviado para a *Cloud*, permitindo que o utilizador possa ter acesso aos mesmos.

Se todos os pedidos, anteriormente referidos, forem recusados, tudo permanecerá inalterado.

3.2 Arquitetura do modelo proposto

A arquitetura foi desenhada de forma que o utilizador pudesse tirar o melhor proveito da rede sem preocupações. Para tal, foi necessário estruturar a melhor forma de como é que a comunicação entre os dispositivos da rede e o *gateway*, pudesse ser realizada, sem que a capacidade interna de cada um não ficasse completamente cheia. Foi feito com que a parte de processamento de dados, fosse realizado pelo dispositivo principal pois, o *gateway*, será apenas um intermediário para o envio de dados para a *Cloud*, pelo que não será necessário sobrecarrega-lo.

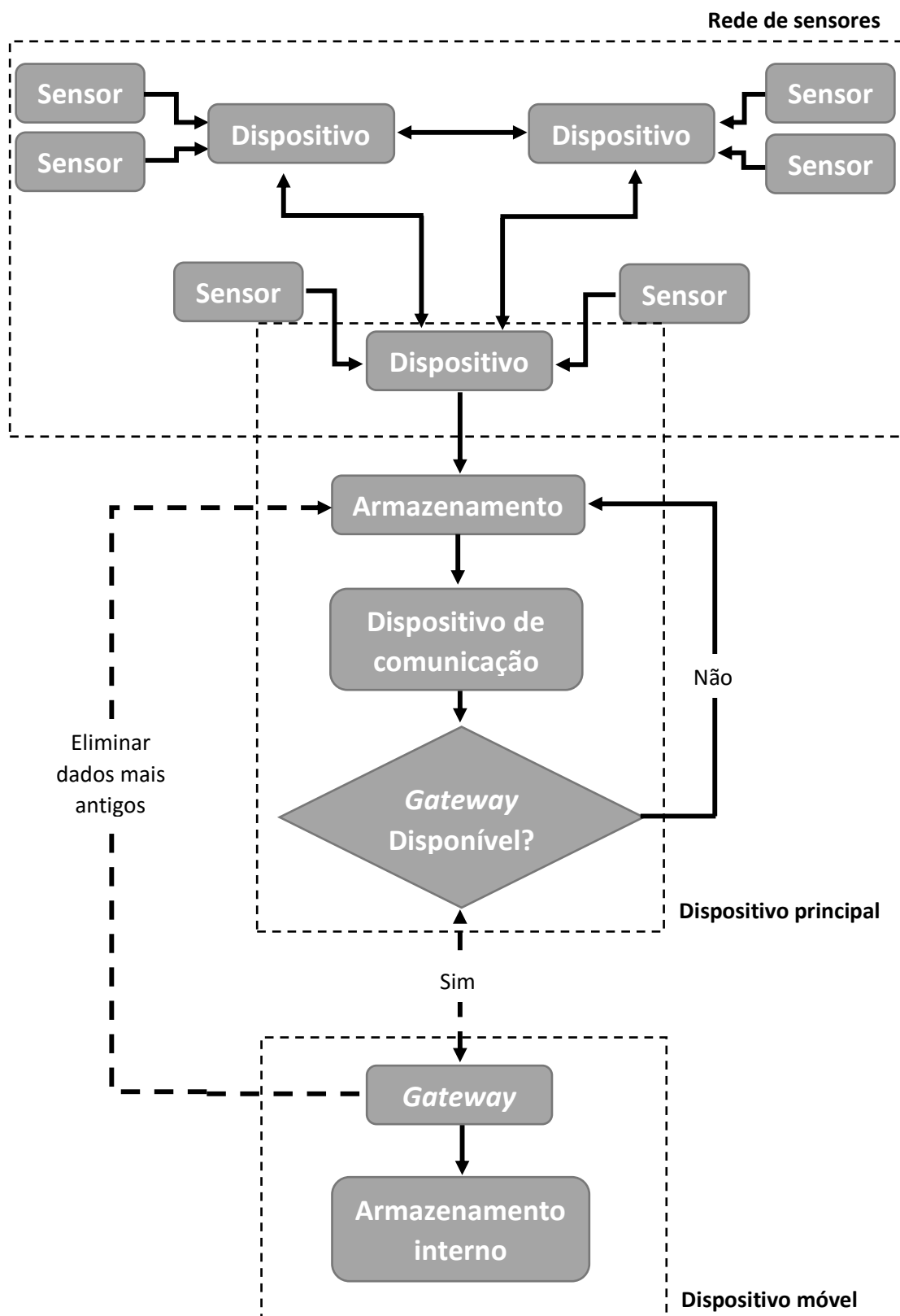


Figura 3.3 – Arquitetura do projeto

De acordo com a figura 3.3, pode ser verificado que o conceito está dividido em três blocos:

- Rede de Sensores – é responsável pelo encaminhamento e recolha de informação. O encaminhamento será feito através de um algoritmo que estabelece uma ligação (mais direta) com o dispositivo principal. A rede de sensores é também responsável por garantir a integridade e qualidade da própria ligação. É composta por três tipos de dispositivos: finais, de encaminhamento e principal. Os sensores estão ligados aos dispositivos finais por cabo. Esses dispositivos finais formam a rede total, que têm como objetivo, fazer com que toda a informação recolhida chegue ao dispositivo principal;
- Dispositivo principal – as suas funções principais são: armazenamento, processamento e emparelhamento com o *gateway*. Este dispositivo principal pertence também à rede de sensores. Toda a informação recolhida pela rede é entregue ao dispositivo principal e armazenada. O armazenamento será sequencial e tem de comprimir a seguinte estrutura: YYYY/MM/DD/HH/nomesensor.txt. A estrutura permite que a procura de todos os ficheiros seja feita de uma forma mais simples. O dispositivo de comunicação está incluído neste bloco, este está responsável pelo emparelhamento do dispositivo principal com o *gateway*
- Dispositivo móvel – um utilizador aleatório poderá participar/ajudar na recolha da informação através do seu *gateway*. Assim, este fica responsável por aceitar o emparelhamento com o dispositivo principal. O *gateway* é também responsável por atualizar todo o armazenamento do dispositivo principal e por atualizar a base de dados na internet. O dispositivo móvel é considerado como transporte de informação temporário, pois, não terá acesso à informação recolhida, nem terá sempre a mesma informação dos sensores.

3.3 Proposta de criação de uma aplicação para PC

Os sensores têm que estar registados nos dispositivos finais, de modo que a informação recolhida seja enviada para o dispositivo principal e, posteriormente para o *gateway*.

A melhor forma é a utilização de uma aplicação que permitirá o utilizador adicionar os novos sensores através do computador. Assim, sem que seja necessário retirar o nó da rede para proceder à sua adição, o utilizador poderá ligar o dispositivo principal ao seu computador e através do executável escolher qual o nó onde foi adicionado o novo sensor.

Em suma o utilizador, com a aplicação instalada no computador, poderá fazer o registo, login e adição de novos sensores. Através do sistema de login, passa também a ter acesso a todos os sensores que tem na sua rede.

4

Validação

Para se proceder à validação do projeto, é necessário verificar quais os fatores e quais as escolhas envolventes que se tomaram em consideração, bem como a sua utilização, para contextualização do mesmo. Posteriormente são apresentados testes de funcionamento do projeto, confirmando assim a sua validação e veracidade de solução (se bem que, com algumas falhas).

4.1 Escolha dos métodos de comunicação

O objetivo desta dissertação contribui para a demonstração de que com este conceito é possível partilhar toda a informação recolhida através de vários sensores e apresentá-los ao utilizador final sem que este necessite de um ponto fixo de internet. Isto é, basta que o utilizador instale os seus sensores num local que deseje, de modo a que quando um outro utilizador (que tenha a aplicação instalada no seu *gateway*) se aproxime do nó principal e partilhe os dados recolhidos desse mesmo sensor.

Para a escolha das tecnologias/módulos é necessário dividir a comunicação em duas partes:

- comunicação entre o nó principal e o gateway;
- a comunicação entre os sensores (sensor principal, encaminhadores e dispositivos finais).

4.1.1 Comunicação do nó principal com o *gateway*

Na comunicação entre o nó principal e o *gateway*, os requisitos que se deve ter em conta são, principalmente, os consumos gerais, tanto a nível de poupança de bateria como custos associados

para o consumidor final (pois, poderia existir uma melhor alternativa, mas poderia ter um custo monetário extra ao cliente, a curto ou longo prazo), facilidade de implementação e instalação; a taxa de transferência de dados não necessita de ser muito elevada pois os dados a enviar não serão de grande dimensão. Foram criadas as tabelas 4.1 e 4.2, baseada em diferentes tecnologias, de forma a facilitar a escolha.

Tabela 4.1 - Tabela de comparação de tecnologias para *gateway*-Arduino (1) ¹

	Bluetooth 4.0 (BLE)	Bluetooth 3.0	NFC	Wi-Fi direct
Ligações em simultâneo	7 ou mais	7	1	1 ou mais
Consumo (mA)	0.4 até 0.8	30	180	0.8 até 215
Segurança	AES 128 bits	AES 8 to 128 bits	-	AES 256 bits
Compatibilidade	Sim	Sim	+/-	+/-
IEEE	802.15.1	802.15.1	-	802.11
Frequência de trabalho (Hz)	2.4G	2.4G	13.56M	2.4G / 5G
Alcance (m)	+/- 100	10 to 100	0.10	200
Velocidade (bps)	2.1M	3 M	424k	1M até 1024M
Preço (\$)	9.95 to 24.95	18.22 to 22.85	29.95	6.95

Tabela 4.2 - Tabela de comparação de tecnologias para *gateway*-Arduino (2) ¹

	Wi-Fi	Infravermelhos	RF
Ligações em simultâneo	1 ou mais	1	-
Consumo (mA)	0.25 até 156	250	18 até 27
Segurança	WEP WPA-PSK WPA2-PSK	-	-
Compatibilidade	Sim	+/-	Não
IEEE	802.11 b/g/n	-	-
Frequência de trabalho (Hz)	2.4G / 5G	115k / 2.4G	434M
Alcance (m)	100 até 400	10	> 150
Velocidade (bits/s)	1M to 300M	-	1k até 128k
Preço (\$)	28.56 até 32.40	0.95	22.95

Analisando as tabelas 4.1 e 4.2, de forma a facilitar o utilizador (sem que tenha que adquirir equipamentos extras para comunicação com os seus sensores) e escolhendo a tecnologia que menos energia consome, foi considerada a utilização do protocolo 802.15.1, ou seja, um BLE

¹ Tabela baseada em produtos em <http://www.sparkfun.com> e <http://pt.farnell.com>

(Bluetooth Low Energy), mas tendo em consideração que nem todos os telemóveis têm compatibilidade com esta tecnologia (4.0), é necessário escolher um módulo que garanta que todos os telemóveis sejam compatíveis com versões anteriores.

O BLE escolhido para este projeto foi o BLE escolhido foi o Tinysine BLE 4.0. Este Bluetooth recebe comandos AT através da interface UART para configuração do mesmo. Este pode trabalhar em modo *Beacon* e *Non-Beacon*.

No modo *Beacon*, o Bluetooth está definido como mestre e aguarda que um outro dispositivo apareça na sua proximidade, enviando um *Beacon-request*, sem superframes. Caso não apareça nenhum outro dispositivo, permanece em modo *Sleep*. Um exemplo recente e comum é a utilização do Bluetooth em modo *Beacon* em supermercados, em que o utilizador aproxima-se de um artigo e o Bluetooth envia uma informação acerca do artigo em questão e termina a ligação.

O modo *Non-Beacon*, o Bluetooth utiliza o múltiplo acesso tradicional, consome mais bateria mas é mais fácil de configurar [64]. De referir que o Bluetooth pode funcionar como mestre e trabalhar em modo *Non-Beacon*, pois neste projeto é pedido que o Bluetooth peça permissão ao utilizador para receber dados dos sensores e não o contrário. As especificações técnicas propriamente ditas serão tratadas no capítulo 4.3.1.

4.1.2 Escolha do equipamento de comunicação da rede em Malha

A topologia Malha foi escolhida de modo a que utilizador possa retirar ou inserir um novo nó na rede. Ou, por exemplo caso extremo, se um dos nós deixa de funcionar, a recolha de informação dos outros nós não fica comprometida, bem como a integridade e o funcionamento normal da rede. Os dispositivos que cumprem os requisitos estão apresentados na tabela 4.3.

Optou-se pelo dispositivo que consumisse menos energia e que garantisse o bom funcionamento da rede, sendo também ele o mais barato quando comparado com as mesmas especificações de outros equipamentos. O dispositivo selecionado foi o Synapse, mais concretamente o Synapse RF200P81. Esta escolha foi baseada no documento em anexo.

O Synapse possui a grande vantagem de ter um microcontrolador capaz de processar scripts na linguagem de programação Python, pois o utilizador pode pedir ao dispositivo que só envie determinados valores, fazendo com que a capacidade do cartão SD não se esgote tão rapidamente. A linguagem de programação Python é bastante simples, de fácil aprendizagem e compreensão e funciona com base na indentação do código. Neste caso o Synapse desenvolveu a sua própria

linguagem de programação, o SNAPpy, que basicamente é uma versão reduzida do Python, facilitando assim o microcontrolador, pois não se torna tão “pesado” o processamento do código.

Tabela 4.3 - Tabela de comparação de tecnologias para a rede em Malha ¹

	ZigBee	Synapse	MI-WI	RF
Ligações em simultâneo	65536	15000 até 150000	-	-
Consumo (mA)	35	23	180	18 até 27
Segurança	AES 128 bits	AES 8 até 128 bits	-	-
IEEE	802.15.4	802.15.4	-	-
Frequência de trabalho (Hz)	868M / 2.4G	2.4G	13.56M	434M
Alcance (m)	> 120	60 até 100	0.10	> 150
Velocidade (bits/s)	250k	2M	424k	1k até 128k
Preço (\$)	17.00 até 32.00	18.22 até 22.85	29.95	22.95

O Synapse tem a particularidade de possuir uma camada de aplicação denominada SNAP, que permite que o utilizador atualize ou que envie comandos *Over-The-Air*. Este pode atualizar todo o script (ficheiros de programação) ou definir no script que, a determinada altura, execute uma função compreendida num determinado nó, através do RPC (*Remote Procedure Call*). Isto permite que, caso exista mais do que um nó com a mesma função descrita no script, seja executada por todos esses nós. O nó que pediu essa execução pode receber também os retornos dessas funções, podendo ser neste caso, os valores lidos pelos sensores. O SNAP funciona através de estados, ou seja, sempre que acontece um evento é disparado um *Hook* que faz com que mude de estado para executar uma ação específica e volta ao seu estado inicial. Por exemplo, no caso de existirem dados no *buffer* do UART é disparado o *HOOH_STDIN*.

No Synapse SNAP o limite de nós utilizados dentro da mesma rede está relacionado com o número máximo de endereços, com a largura de banda da camada física e com a segmentação da rede. Cada SNAP tem o seu endereço, que corresponde ao endereço MAC IEEE. Este equivale a um único número de 64bits, isto é, pode existir 1.8×10^{19} números diferentes que identifiquem cada um dos dispositivos SNAP.

A limitação da largura de banda disponível está relacionada com a capacidade com que a camada de rede tem de transportar os dados a enviar. No caso desta dissertação, que porventura, é também o exemplo fornecido pelo *White-Paper* (tabela de especificações), se um módulo trabalhar a 2.4GHz e a uma taxa de transferência de 2Mbps:

¹ Tabela baseada em produtos em <http://www.sparkfun.com> e <http://pt.farnell.com>

$$\frac{2000000bits/sec}{8bits/byte} \times \frac{1pkt}{128bytes} = 1953 pkt/sec$$

que na pior das hipóteses cerca de 25% desse valor fica disponível para processamento e para a transmissão dita em si, então é dado que são enviados cerca de 1500 pacotes por segundo (pps – packets per second). Por exemplo, no caso de a rede se expandir muito e tenha que enviar dados, como por exemplo, a cada 10 segundos, então significa que a rede pode-se expandir até $1500 \times 10 = 15000$ dispositivos, antes de chegar à largura de banda disponível anteriormente calculada.

Existem diferentes opções de contornar este crescimento como:

- Alterar para uma largura de banda maior na camada física
- Diminuir a taxa de transferência dos nós
- Segmentar a rede

Os encaminhadores dos dispositivos Synapse permitem que sejam utilizadas diferentes larguras de banda na comunicação entre dois dispositivos. Como o Synapse pode ser programado, pode ser pedido que o Synapse só envie dados caso o valor lido pelo sensor tenha sido alterado em 5%. Isto pode implicar que a rede possa crescer até 10 vezes mais, ou seja, 150000 dispositivos. A parte da segmentação da rede facilita pois, em vez de serem utilizados como no caso anterior, 150000 dispositivos no mesmo canal, para uma melhor eficiência, estes dispositivos podem muito bem serem divididos por canais diferentes.

O Synapse possui uns parâmetros NV (*Non-Volatile*) na sua memória que estão disponíveis para o utilizador, mas alguns deles estão reservados para sistema em si, ficando assim disponíveis 116 parâmetros. Estes parâmetros ficam sempre gravados mesmo que o dispositivo reinicie ou fique sem bateria. Como já tem os parâmetros anteriormente indicados e uma máquina virtual incorporada no dispositivo, o Synapse tem que impor algumas restrições, como é o caso da versão reduzida do Python, e possui também um limite máximo de variáveis, que são 8 *Strigns* de 126 caracteres [65]. É referido bastantes vezes nesta dissertação que existe um nó principal, nós finais e encaminhadores, mas o que são concretamente?

O nó principal (foi tratado como nó para tratar o dispositivo Synapse e os sensores a ele ligados como um todo, para facilitar a compreensão da dissertação) é considerado como principal pois é o dispositivo que está diretamente ligado ao Arduino e também porque o Synapse não tem um dispositivo coordenador [66]. Os nó final podem também ser encaminhadores dependendo da ação que estiverem a realizar, como por exemplo, um dispositivo final pode estar frequentemente a fazer leitura dos sensores mas, se por algum motivo, um outro dispositivo na rede deixa de

funcionar, este pode fazer de encaminhador de forma a fornecer um caminho para que os outros nós finais enviem os seus dados.

4.1.3 Componentes utilizados

Em primeiro lugar, é necessário uma placa onde se possa adicionar o Synapse RF, de forma a facilitar a adição dos sensores, bem como a fonte que o irá alimentar. Neste caso o componente escolhido foi um adaptador Synapse-XBee e um módulo (XBee) de adição para a bateria como mostra a figura 4.1. Os indicadores LEDs podem não ser inseridos, caso se queira que a bateria dure bastante tempo, pois só servem para indicar se o Synapse está a enviar ou a receber dados. Estas adaptações são feitas pois como não existe muita informação acerca do Synapse, os equipamentos necessários para o mesmo também não existem.



Figura 4.1 - Adaptadores para o Synapse [67] [68]

É também necessário um módulo onde se possa adicionar o Synapse principal e o cartão no mesmo módulo. Foi tido em conta esta opção para que não seja necessário passar fios de um módulo para o outro, o que implicaria mais perdas, logo menos tempo de bateria útil. O módulo encontrado é uma adaptação para o XBee indicado na figura 4.2.

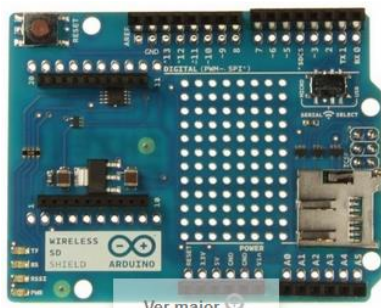


Figura 4.2 - Shield para Arduino, Synapse e Cartão SD [69]

4.2 Comunicação através do Arduino

O Arduino tem um foco bastante grande nesta dissertação, pois é através dele que passa toda a informação, onde fica armazenada, onde é gerenciado os ficheiros e onde é feita a comunicação

com o telemóvel. É também através dele que o utilizador regista novos sensores, ligando o Arduino ao computador. O Arduino é utilizado pois é bastante fácil de programar o microcontrolador (que posteriormente pode ser retirado e usado sem nenhum auxílio do Arduino).

O Arduino fará o controlo dos dados recebidos através da rede em Malha e, posteriormente, o envio desses dados para o telemóvel (quando possível). Na primeira configuração irá criar um ficheiro `infoSensores.txt`, que terá a informação do utilizador como, o nome de utilizador e uma palavra passe (encriptada) escolhida pelo mesmo.

4.2.1 Informação recebida da rede em Malha

O Synapse, por si, faz o controlo do fluxo de mensagem o que implica que o Arduino, através de uma porta Serial, ligada ao Synapse, irá receber as mensagens sequencialmente.

Inicialmente, o Aduino aguarda até que exista alguma mensagem no Serial vindo do Synapse. Assim, o Arduino vai receber uma *String*, criada pelo Synapse que como está em formato JSON (formato de interligação de dados) é fácil de aceder aos campos da mensagem individualmente. É retirado da *String* o ano, mês, dia, hora e nome do sensor pois será nesta ordem que a diretoria do ficheiro vai ser criada e guardada no cartão SD (por exemplo `2015/6/21/17/TempCasa.txt`) O nome do sensor poderá ter somente 8 caracteres, visto que a biblioteca do Arduino só deteta o nome dos ficheiros até esse tamanho. Por exemplo, se existisse um nome de um ficheiro `TemperaturaCasa.txt`, o Arduino só iria apresentar `Tempera~.txt`. Os ficheiros são criados de hora a hora, para facilitar a procura dos mesmos, no cartão SD.

Quando um sensor é adicionado, é necessário estabelecer a conexão do Arduino com o computador e, desta forma, o utilizador pode registar um novo sensor, através da aplicação. Este tem que preencher os campos: nome do sensor, nó onde adicionou o sensor e porta do nó. O processo fica ao cuidado do utilizador pois não existe nenhuma proteção através da qual verifica se o nó indicado pelo utilizador será o correto ou não. Essa informação é recebida pelo Arduino e enviado, sem nenhum processamento para o Synapse.

4.2.2 Informação enviada por Bluetooth para o telemóvel

Existem duas portas Serial, uma para o Synapse, e outra para o Bluetooth.

O Bluetooth vai estabelecer a conexão com o telemóvel para permitir que os dados sejam recebidos e enviados para a *Cloud*. Foram previamente definidos códigos de comando para facilitar a comunicação:

- “OK+CONN” – Este código é escrito pelo próprio Bluetooth que está anexado ao Arduino. Serve para que o Arduino saiba que o telemóvel e o Bluetooth já estão emparelhados.
- “OK+LOST” – Este código é escrito pelo próprio Bluetooth que está anexado ao Arduino. Serve para que o Arduino saiba que a comunicação entre o telemóvel e o Bluetooth terminou.
- “Pass” – Palavra passe enviada pela aplicação instalada no telemóvel, impedindo que quem não tenha a aplicação, não consiga ter acesso aos dados. Esta palavra passe é encriptada de modo a garantir o máximo de segurança.
- “NewDate” – É enviado pela aplicação do telemóvel a data mais recente e registada no momento da ligação do telemóvel com a base de dados, ou seja, quando o utilizador do telemóvel se conectou a internet, a aplicação acedeu à base de dados verificando qual a data mais recente inserida.
- “ReceiveData” – É enviado pela aplicação do telemóvel um pedido de envio de dados, ou seja, é pedido ao Arduino que envie os dados de um sensor para o telemóvel.

O Arduino, tanto na comunicação com o Bluetooth como na comunicação com o Synapse, está sempre em funcionamento, de modo a estar sempre disponível quando um telemóvel tenta estabelecer uma ligação entre eles. Ou seja, quando um telemóvel se aproxima, e que esse mesmo telemóvel tenha a aplicação instalada e o Bluetooth disponível, o Arduino tenta estabelecer a ligação com a aplicação. Se aplicação não estiver instalada e, mesmo assim, existir uma tentativa de emparelhamento, o Arduino irá automaticamente terminar a ligação, pois a palavra passe de confirmação não foi enviada para o Arduino de forma a desbloquear a comunicação. Neste projeto, como foi utilizado um dispositivo semelhante ao escolhido, a comunicação entre o telemóvel e o Arduino só é realizada caso o utilizador do telemóvel faça um pedido de emparelhamento. Caso fosse exatamente o dispositivo escolhido este, quando em modo mestre, sendo necessário introduzir os comandos OK+INNE 1 e OK+ROLE 1, fica então responsável pelo emparelhamento entre os dispositivos.

O estado inicial do Bluetooth é em modo *Sleep* (adormecido) que é acordado através da porta Serial do Synapse de forma a estarem em sincronismo (visto que o Synapse possui *Real Time Clock*), ou seja, a cada 10 segundos, o Bluetooth é acordado, de modo a procurar dispositivos na sua proximidade. Caso não exista uma ligação bem-sucedida, volta para modo *Sleep*, disponibilizando que o Synapse guarde as informações recolhidas pelos sensores no cartão SD. O Bluetooth quando em modo Ativo, “ocupa” o Arduino, pois necessita de acesso ao cartão SD, caso a ligação seja bem-sucedida.

No caso de emparelhamento com o telemóvel, o Arduino aguarda o comando “NewDate”. Este comando serve para informar o Arduino, que apague os dados mais antigos que estejam, garantidamente, na base de dados na internet. Quando o utilizador liga o telemóvel à internet a aplicação irá verificar através de um serviço web, qual a data mais recente existente na base de dados de um sensor específico e essa informação fica guardada no telemóvel. Por opção, o telemóvel não pode guardar mais de vinte sensores de uma vez, de modo a não comprometer a capacidade interna do telemóvel. Após o telemóvel do utilizador ter essa informação, quando estiver próximo de um nó e se conectar ao Arduino, vai ser enviada por Bluetooth (quando recebido o comando “NewDate”) a informação dos vinte sensores. Assim, o Arduino vai procurar no cartão SD se existem ficheiros com datas anteriores à data recebida e caso positivo apaga os ficheiros que contém os dados desse sensor específico. Caso contrário envia uma mensagem para o telemóvel de forma a indicar que toda a informação até à data já foi removida do cartão SD com sucesso. Caso nenhum dos vinte sensores pertencer à rede em que o utilizador se conectou, não é feita nenhuma verificação e tudo permanece inalterado.

Posteriormente o telemóvel envia o comando “ReceiveData” para informar o Arduino que o telemóvel está pronto para receber novos dados dos sensores. O Arduino, através da variável “oldDate” (esta variável é atualizada sempre que esta função é executada) e caso já exista uma data guardada nessa mesma variável, o Arduino tenta abrir o ficheiro que tenha essa data, que em caso de falha (pois a função anterior “NewDate” poderá já ter apagado o ficheiro) é procurado o ficheiro seguinte mais antigo. Depois de aberto o ficheiro com essa data, é iniciada a transferência de dados com o telemóvel. A transferência é executada, até o ficheiro completo ter sido enviado. Os ficheiros são criados de hora a hora, pelo que só podem ser enviados no máximo sessenta dados dos sensores, caso se assuma que os sensores retirem informação de minuto a minuto.

Quando nenhum utilizador se aproxima do nó principal, por consequência, o cartão SD começa a ficar sem capacidade. Assim, é gerada uma mensagem (“CardFull”) de modo a que sejam desligados os nós finais, impedindo que recolham mais informação. Quando uma pessoa se aproximar do

sensor é então enviada essa mensagem (em vez do ficheiro com as informações recolhidas pelos sensores) de modo que o utilizador daquela rede de sensores seja notificado.

Após a confirmação na base de dados de que o cartão SD está cheio, o nó principal recebe uma mensagem de forma a desliga-se a ele próprio, aguardando que o utilizador vá recolher a informação dos sensores. Este método permite que as baterias dos sensores não sejam gastas indevidamente. Depois da recolha dos dados o utilizador tem de reiniciar o sistema.

O esquemático de todo o sistema visto do Arduino é indicado na figura 4.3.

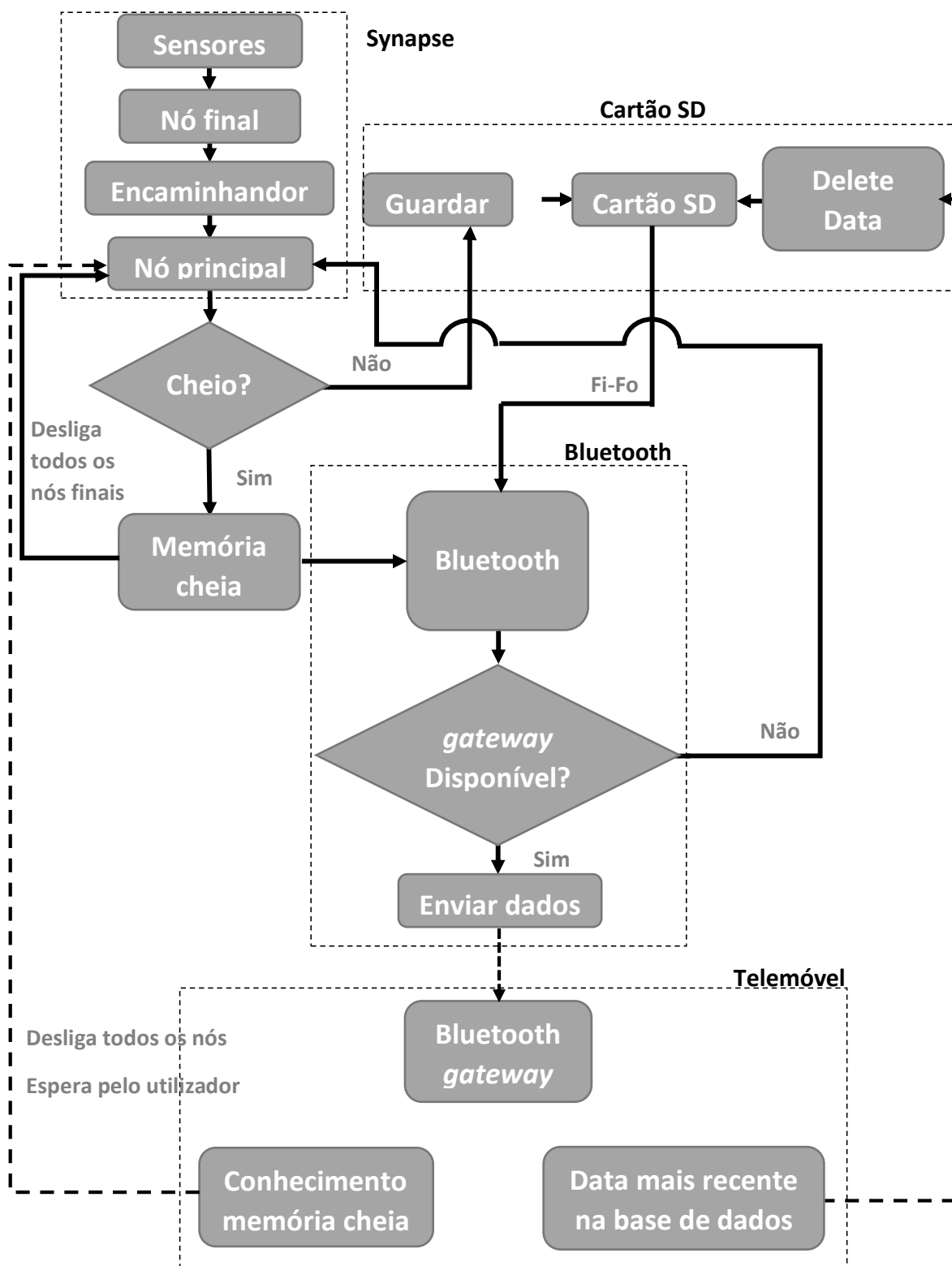


Figura 4.3 - Fluxograma visto do Arduino

4.3 Programas desenvolvidos em Python

No desenvolvimento da aplicação para registo, controlo de novos utilizadores e sensores, o utilizador tem de possuir computador com ligação à Internet e com pelo menos uma ligação USB.

4.3.1 Interface com o utilizador – Registo

Na solução proposta, o utilizador tem que estar registado para poder adicionar sensores à sua rede. Foram criadas duas formas de registo do utilizador, uma via *Web Site* (desenvolvida e criada na dissertação Ponto de Acesso Móvel em Ambiente Sensorial) e outra através de uma aplicação, instalada pelo utilizador no seu computador.

Esta aplicação (figura 4.4) foi desenvolvida em linguagem de programação Python por ser uma linguagem bastante compreensível e de fácil implementação. Existem duas versões do Python, o Python 2 e o Python 3. Ambas têm diferentes formas de programar, tendo sido escolhida a versão 2 pela vasta existência de bibliotecas e bom feedback por parte dos utilizadores.

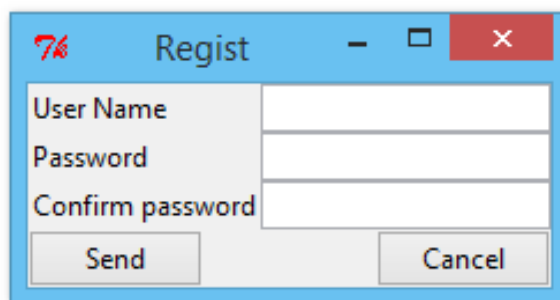
A imagem mostra uma janela de software com o título 'Regist'. No canto superior esquerdo, há um ícone de uma seta vermelha apontando para cima. A janela contém três campos de texto rotulados 'User Name', 'Password' e 'Confirm password'. Abaixo dos campos, há dois botões: 'Send' à esquerda e 'Cancel' à direita. A janela tem uma barra de título azul com botões de minimizar, maximizar e fechar.

Figura 4.4 - Registo de um novo utilizador

Para executar um novo registo é verificado através de uma tentativa de contacto com um serviço Web, ou seja, através de uma biblioteca denominada *Suds web services client*, que é baseado em cliente SOAP, se o utilizador está conectado à internet. Este cliente, quando criado, serve para que o WSDL (conhecendo o URL dos serviços web e os seus serviços) seja processado. Quando é feita uma tentativa de registo é verificado se esta ligação foi bem-sucedida (assumindo que o servidor nunca terá falha), e assim, assume-se que o utilizador tem ligação à internet, caso contrário é pedido que verifique a sua conexão.

Posteriormente, é feita uma pergunta ao serviço Web, em que a mesma é um serviço que tem que ser previamente reconhecido, incluindo também os parâmetros que enviam a informação necessária para o registo do cliente. Após a execução desse serviço, é devolvida uma resposta em

caso de sucesso ou fracasso. Na figura 4.5 é apresentando um exemplo de um pedido a um serviço web.

```
if client.service.register_new_user(topLevel.userName.get(),
                                     hashlib.sha512(topLevel.password.get()).hexdigest(),
                                     'true','normal') == 'registadocomsucesso':
    tkMessageBox.showinfo(info,"Successfully Registered")
    self.onClose()
else:
    tkMessageBox.showinfo(info,"Error creating new user")
    return
```

Figura 4.5 - Pedido a um serviço web

Quando é enviada a pergunta para pedido de serviço, é também enviado a palavra passe do utilizador, pelo que a mesma tem que estar encriptada e não pode ficar armazenada em nenhuma variável do programa de modo a garantir a segurança dos dados do utilizador. Esta é encriptada com o algoritmo SHA-512.

O registo pode ser executado somente para um utilizador particular ou para uma empresa (organização), como por exemplo, a FCT. Esta pode querer registar-se e ter os seus próprios sensores mas permitir que os seus alunos tenham acesso a eles.

4.3.2 Login e registo de sensores

Após o registo do utilizador, este pode começar a registar os seus sensores. Para que tal seja possível, o utilizador tem que fazer o *Login*. Este pode ser feito tanto no site como na aplicação.

O Login para registo de novos sensores tem que ser feito através da aplicação, pois é necessário que esta envie dados para o Arduino como, o nome do sensor, o endereço do Synapse a que pertence esse novo sensor, a porta (de referir que é considerado porta, um input com dois pinos, um pin ground e outro ligado ao dispositivo Synapse) onde foi o sensor foi adicionado.

O utilizador, antes de fazer o *Login* tem que escolher se pertence a um particular ou a uma organização e, da mesma forma que o registo, é enviado uma pergunta ao serviço Web para execução do serviço de *Login*. Também aqui a palavra passe do utilizador é enviada pelo cabeçalho (*headers*) com encriptação em SHA 512. Os cabeçalhos assemelham-se aos parâmetros de entrada das funções de programação, como por exemplo, a função(int a, int b) tem como parâmetros de entradas as variáveis a e b.

```
client.set_options(headers={'username':self.userName.get(), 'password':hashlib.sha512(self.password.get()).hexdigest()})
```

Após o *Login*, em caso de sucesso, o utilizador tem agora acesso a todos os seus sensores. Por opção, é utilizado uma biblioteca denominada JSON que basicamente é um formato de interligação de dados (ou de texto) de fácil leitura e escrita, criando uma lista de valores ordenados. Com esta biblioteca a lista de sensores contida na base de dados, em formato JSON, é passada para uma variável que posteriormente se consegue aceder a qualquer parâmetro dessa mesma variável. Um exemplo de uma lista em JSON é:

```
{'type_sensor': 'Temperatura',  
'owner_id': '0873558c7e8441ad8d93ae3b944c8d45',  
'last_communication': '2015/03/02',  
'device_id': '321'},  
{'type_sensor': 'Humidade',  
'owner_id': '0873558c7e8441ad8d93ae3b944c8d45',  
'last_communication': '2015/03/02',  
'device_id': '123'}}
```

No exemplo acima apresentado, mostra que o utilizador possui dois sensores, um de Temperatura e outro de Humidade, e que estão inseridos em dispositivos diferentes. Como é possível verificar, a identificação do utilizador (*owner_id*) está encriptado.

Existem três tipos de Login, o Login particular (como anteriormente explicado), o Login de uma organização e o Login de um utilizador pertencente à organização.

No caso de uma Organização, para além de serem apresentados os sensores é também apresentado os utilizadores que pertencem à organização. Neste caso somente a organização pode adicionar utilizadores e sensores.

No caso de Login de um utilizador pertencente a uma organização, o Login tem que ser feito como se fosse um particular pois, o utilizador, vai ter acesso aos sensores da organização mas não possui nenhum privilégio para além desse.

Para adicionar um novo sensor, o Arduino tem que estar ligado por USB ao computador e definir a porta do sensor, o nome do novo sensor e o endereço do Synapse onde foi adicionado o sensor. Após a aplicação detetar o Arduino, é verificado na base de dados (através de um serviço) se o novo nome que o utilizador definiu para o seu sensor já existe na sua rede de sensores. Posteriormente é verificado pelo Arduino se a porta do Synapse onde o Sensor adicionou esse novo sensor está

disponível. Em caso de sucesso, é adicionado o nome do sensor para a ROM do Synapse, de modo que se saiba quais os dados pertencentes ao mesmo. É também adicionado o nome, novamente através de um serviço, na base de dados.

Este registo interliga os 4 dispositivos que são utilizados na dissertação, ou seja, o computador, o Arduino, o Synapse e o sensor, e todos comunicam pela porta serial. Quando o utilizador carrega no botão “Registar” da aplicação, é enviado para o Arduino a palavra passe definida pelo utilizador que faz com que este entenda que se está a tentar registar um novo sensor, o que vai fazer com que a rede de sensores pare por uns segundos até o novo sensor se adicionado corretamente.

4.3.3 Comunicação entre Arduino e Synapse

Depois do Arduino obter o nome, a porta e o endereço onde o novo sensor se encontra, a comunicação entre o computador e o Arduino termina, e é enviado essa informação para o Synapse, novamente através da porta serial. Este código vai informar o Synapse que existe um novo sensor a ser adicionado e, essa informação vai ser guardada na memória interna do mesmo. Sempre que a rede tiver um problema e tenha que ser reiniciada, não será necessário configurar tudo de novo.

O Synapse tem a possibilidade de fazer atualizações de todo o script ou enviar dados de dispositivo para dispositivo através da própria rede (*Over-The-Air*), facilitando assim o utilizador, pois não necessita de retirar um nó completo da rede para adicionar um novo sensor. A função que permite adicionar informação na memória interna é *“saveNvParam(id,obj)”* em que o id é o parâmetro onde se quer adicionar informação. Existem parâmetros “id” (ID 128 até ao ID 254) livres que servirão para guardar a informação dos sensores onde o “obj” será o nome do sensor. É possível também alterar o nome do dispositivo Synapse permitindo assim que o utilizador possa inserir o nome do mesmo e não o endereço do Synapse, esse nome pode ser alterado através do parâmetro ID 8.

Através do RPC e Callback (tem o mesmo conceito que o RPC, sendo que a diferença é que no Callback, as funções “chamadas” retornam valores, e o RPC apenas executa as funções), o nó principal vai enviar um comando através da rede de modo que seja executado por todos os nós uma função específica. Neste caso, vão existir duas únicas funções específicas a serem executadas pelos nós finais são:

- *“newSensor(pin,name)”* – servirá para guardar o nome do novo sensor adicionado;

- *“callback(‘saveCard’, ‘sendData’)”* – servirá para enviar os dados lidos pelos sensores para o nó principal.

As funções são executadas somente pelo nó principal que, no caso da função *“readArduino(data)”*, só é executada quando existe um novo sensor a ser adicionado. Mais especificamente, esta função é disparada pelo *HOOK_STDIN*, visto que cada vez que surge uma mensagem na porta UART esta função é executada. Esta recebe principalmente a informação do novo sensor adicionado através do Arduino, sendo que por vezes pode também receber outras funções que o utilizador pretenda executar. Para tal, o Arduino envia uma mensagem em que o primeiro parâmetro é uma opção (recebido com inteiro). Caso a opção seja igual a:

- 1 – indica que a mensagem recebida será, respetivamente, o endereço do Synapse onde se pretende adicionar o sensor, a porta e o nome do sensor;
- 2 – indica que o utilizador pretende que o Synapse de destino passe a fazer parte de um outro endereço de rede por ele definido, em que a mensagem recebida será o endereço do Synapse de destino e o novo endereço;
- 3 – indica que o utilizador pretende alterar o canal de comunicação do Synapse de destino, sendo a mensagem recebida igual à opção anterior, que ao invés de receber o novo endereço recebe o novo canal
- 4 – indica que o utilizador quer alterar o nome do Synapse de destino, novamente a mensagem recebida pelo Arduino será igual à opção anterior, e recebe o nome que pretende para o Synapse.
- 5 – permite que o utilizador faça com que o Synapse específico reinicie, recebendo uma mensagem somente com a opção e o endereço do Synapse de destino.

Todas estas opções são executadas através de RPC, que por exemplo, caso a opção seja igual a um, o Synapse principal vai executar a função, *“rpc(endereço_Synapse, ‘newSensor’, porta, nome)”*. O *“newSensor”* é a função a ser executada no sensor de destino. Nas seguintes opções é executada uma função já pré definida que não precisa de ser criada no script, que é o *“saveNvParam”*, que vai fazer com que os parâmetros sejam automaticamente guardados na memória.

Os nós finais, só funcionam caso o nó principal peça para estes executarem uma função. O nó principal quando pede para o nó final adicionar um sensor, este verifica na memória, através da função *“loadNvParam(pin+128)”* se já existe algum sensor adicionado a porta pretendida (apesar de ser redundante, pois essa comparação já é feita anteriormente quando o utilizador regista o sensor, mas poderá ser visto como uma garantia que não vão ocorrer erros) e, caso não exista, os

pinos são ativados como Input/Output. A variável `pin`, é um inteiro referido pelo utilizador que indica o parâmetro de memória onde vai ser guardado o nome do sensor (ou seja, se o utilizador adicionar um sensor na porta 0, então o seu nome vai ser guardado no ID 128 da memória). Depois, através da função `"saveNvParam(pin+128,name)"` o nome do novo sensor é adicionado no parâmetro de memória correto, para próxima verificação.

Quando o nó principal pedir aos nós finais que enviem a informação recolhida de todos os sensores, então estes verificam novamente os parâmetros de memória para saber quais as portas que estão ativas e assim, agrupa toda a informação numa *String*: {o que leu, nome do sensor}, novamente em formato JSON. Posteriormente é retornada essa *String* para o nó principal e é adicionado a data e hora a que foi feita essa leitura, ficando a *String* agora com o formato: {ano,mês,dia,hora,minuto,segundo},{o que leu, nome do sensor}.

Por fim quando a configuração dos sensores ficar concluída, o nó pode então iniciar a recolha dos dados dos sensores e envia essa informação para o nó principal. Quando esses dados chegam ao nó principal é necessária a comunicação com o Arduino para guardar tudo no cartão SD.

A opção tomada implica dois fatores críticos: em primeiro lugar o congestionamento de mensagens caso os nós tenham muitos sensores adicionados, que vai atrasar a recolha dos dados e um gasto maior de energia por parte do nó principal; o segundo fator deve-se, por tornar o tempo de leitura de todos os sensores todos iguais, ou seja, um sensor de temperatura e um sensor de deteção de monóxido de carbono, não necessitam de estar a recolher informação com a mesma frequência.

Foi tomada esta decisão para simplificação do projeto envolvido, mas a forma de contornar estes fatores passa por aproveitar também a memória do próprio nó e criar uma condição de tempo, ou seja, quando o nó principal pedisse a todos os nós que enviassem os dados recolhidos, era primeiro verificado quais os sensores que teriam que enviar a sua informação naquele tempo antes de ser enviado os dados.

4.4 Consumo e especificações

Os sensores, quando ligados a um dispositivo que permita o envio dos seus dados através de redes sem fios ou mesmo por cabo, passa por diferentes tipos de estados, Ativo, "TX", "RX", *Idle* e *Sleep* (modo de processamento, envio de dados, receção de dados, descanso e adormecido).

A Transmissão, Recepção e *Idle*, são os estados que mais energia consomem, para tal, é também necessário uma otimização do tempo de sincronismo, pois o tempo de espera em recepção consome uma elevada quantidade de energia. Outra forma de consumo desnecessário de energia é quando o sensor está em modo recepção e tenta receber informação quando não existe ninguém a enviar, ou quando tem excesso de informação transmitida (*overhead*).

Neste projeto existem três dispositivos que consomem mais energia: o dispositivo Bluetooth, o Arduino e o Synapse.

Para uma estimativa da durabilidade das baterias, é necessário ter em conta as seguintes expressões [70]:

- Tempo de transmissão (t_x) – Esta expressão, como o nome indica, vai dar o tempo (em ms) que uma transmissão leva a enviar uma certa mensagem, esta depende do tamanho da mensagem, n (em bytes), do tamanho do endereço do dispositivo (*MAC address*) e da taxa de transmissão, dr (em Kbps);

$$t_x = \frac{8 \times (MAC + n)}{dr} \quad (1)$$

- Tempo em que o dispositivo está ativo (t_{act}) – Nesta expressão, está incluído o tempo (em ms) que o dispositivo leva a reconhecer o nó principal, o tempo de resposta dos dispositivos e o tempo de transmissão;

$$t_{act} = t_{on} + t_x \quad (2)$$

- Consumo médio em que o dispositivo está ativo (I_{act}) – O consumo médio (em mA) depende das expressões 1 e 2;

$$I_{act} = \frac{t_{on} \times I_{on} + t_x \times I_x}{t_{act}} \quad (3)$$

- Corrente de descarga (I_{drain}) – Esta corrente (em mA) depende do tempo e do consumo médio em que o dispositivo está ativo, somado ao tempo e consumo médio em que o dispositivo está em modo *Sleep*, em que T é o período entre duas transmissões;

$$I_{drain} = \frac{t_{act}}{T} \times I_{act} + \left(1 - \frac{t_{act}}{T}\right) \times I_{Sleep} \quad (4)$$

- Durabilidade das baterias em anos (Tempo) – Assumindo uma bateria com capacidade C (em mAh) e utilizando a formula anterior pode-se dizer que:

$$Tempo = \frac{C/(365 \times 24)}{I_{drain}} \quad (5)$$

4.4.1 Bluetooth

Não foi possível a utilização do Bluetooth escolhido da TinySine (figura 4.6), pelo que se teve de optar por um Bluetooth equivalente com praticamente os mesmos consumos e *DataSheet*.



Figura 4.6 - Bluetooth TinySine [71]

Este trabalha a uma frequência 2.4GHz e tem uma taxa de 21Mbps que para este caso não era necessário, pois os dados a enviar são bastante pequenos (aproximadamente 4KB) e a velocidade da comunicação não é muito importante. Tentou-se, de certa forma, que as tensões de alimentação fossem relativamente baixas de modo a permitir o uso de fontes externas, sem ser necessário eletrónica de potência, pelo que o Bluetooth trabalha a 3.3VDC a 50mA. Já a potência da sua antena era um fator importante e pode ser alterada com os comandos AT utilizados para configurar o Bluetooth, pelo que pode ir de 0.01mW até 5mW o que vai proporcionar um alcance em espaço aberto até 100m. Tem a vantagem de a sua comunicação ter um algoritmo de encriptação AES 128 bits que ajuda na segurança dos dados do utilizador [71].

O Bluetooth vai alterando os seus estados consoante a recolha dos dados do Synapse.

Cada vez que o Bluetooth está no seu estado ativo, consome 8.5mA e quando altera o seu estado para *Sleep* consome entre 400µA e 1.5mA, o que é bastante bom pois, assumindo que o Bluetooth

vai procurar dispositivos a cada dez segundos (visto que se for uma zona mediantemente povoada convém que o Bluetooth esteja disponível para enviar os dados sempre que possa), significa que durante esses dez segundos está em modo *Sleep* e depois altera o seu estado para Ativo durante um segundo para fazer a procura de dispositivos:

$$\text{consumo médio} = \frac{8.5m \times 1s + 400\mu \times 10s}{1s + 10s} \approx 0.77mA$$

Assumindo agora que o Arduino estabeleceu uma ligação com um telemóvel e assumindo que tem um ficheiro de aproximadamente 4KB, ou seja, 32Kbits para enviar para o telemóvel então, se o Bluetooth é capaz de enviar 21Mbits por segundo e pela expressão (1):

$$t_x = \frac{8 \times (12 + 4000)}{21000} = 1.53ms$$

Assim era possível enviar o ficheiro em 1.53ms mas, no entanto, foi medido uma taxa de envio de dados de 9600bps pois é necessário aceder ao ficheiro que está no cartão e, o leitor de cartões adquirido não é o melhor em termos de velocidade de acesso, então:

$$t_x = \frac{8 \times (12 + 4000)}{9.6} = 3343.3ms$$

Pelo que o Bluetooth tem de estar a transmitir durante, aproximadamente, 3.3 segundos. Adicionando agora o tempo que leva a estabelecer a ligação e o tempo total de todo o processo entre o Bluetooth e o telemóvel explicado no capítulo 4.1, o Bluetooth quando estabelece uma ligação leva entre 4 a 5 segundos a concluir todo o processo, isto dependendo do tamanho do ficheiro. Como foi dito anteriormente, o Bluetooth a cada dez segundos verifica durante um segundo se existe um Bluetooth na proximidade, assim em uma hora, significa que 360 segundos são gastos para tal. Portando pela expressão (2), (3) e (4), e supondo que a cada hora existe uma conexão entre o Bluetooth e o telemóvel:

$$t_{act} = 360000 + 3343.3 = 363343.3ms$$

$$I_{act} = \frac{360000 \times 1.5 + 3343.3 \times 8.5}{360000 + 3343.3} = 1.56mA$$

$$I_{drain} = \frac{363343.3}{3600000} \times 1.5 + \left(1 - \frac{363343.3}{3600000}\right) \times 0.0004 = 0.158mA$$

Verificando assim que o Bluetooth tem uma corrente de descarga, na bateria, de 0.158mA, que como vai ser apresentado de seguida, é bastante superior à corrente de descarga feita pelos dispositivos Synapse.

4.4.2 Synapse



Figura 4.7 - Synapse RF200P81 [72]

O Synapse RF200P81 (figura 4.7) trabalha com uma frequência igual a 2.4GHz, tem vinte pinos GPIO em que 8 deles podem ser definidos com PWM, pelo que podem ser adicionados até dez sensores. Funciona de 1.8V a 3.6V dependendo do modo que está a ser utilizado, mas tipicamente funciona nos 3.3V, como uma taxa de transferência de 2Mbps. Este tem um alcance que depende da sua taxa de transferência (250Kbps, 500Kbps, 1Mbps e 2Mbps) que pode variar entre 450 a 750 metros. Também, este tem um microcontrolador ATmega128RFA1 e uma memória *flash* de 128k, em que 58.5k são livres para que o utilizador do Synapse possa fazer *upload* dos *scripts* criados pelo mesmo.

A sua comunicação também já possui um algoritmo de encriptação AES-128 nas mensagens enviadas pela rede, garantindo assim uma melhor segurança no envio dos dados.

Para além da memória *flash*, para atualização dos *scripts*, tem uma memória EEPROM de 4K para guardar dados que sejam necessários.

A comunicação também lida com os erros que possam ser criados durante o envio de dados, como é o caso da mensagem não chegar ao destino corretamente. Assim o Synapse usa um método de reenvio e reconhecimento (*Retries and acknowledgement*), que quando uma mensagem é enviada para outro Synapse este aguarda até receber um reconhecimento de que a mensagem chegou corretamente. Caso isso não aconteça, é enviado uma nova mensagem, e assim sucessivamente até este ter o reconhecimento da mensagem [72].

Tal como no Bluetooth, o Synapse, de forma ter uma gestão boa da bateria também tem um modo *Sleep*. Na rede de sensores, o nó principal será o que menos tempo estará nesse modo, pois tem que receber os dados de todos os outros nós e ainda comunicar e enviar essa informação para o Arduino. A diferença é que, quando envia ou recebe dados, o Synapse também coloca o CPU em funcionamento para executar os scripts criados.

Neste caso é um pouco complicado estimar quanto será o consumo do dispositivo, pois depende do algoritmo que tem da rede em Malha, pois o nó pode ser constantemente utilizado como encaminhador de outro nó e recolher informação dos sensores que possui, ou pode somente ser utilizado para recolha de informação.

Supondo um nó final, que tem dez sensores inseridos pelo utilizador, e nunca fará de encaminhador, irá ter que enviar esses mesmos dados uma vez a cada minuto (tempo esse definido por pré definição, utilizado para testes). O nome do sensor poderá ter no máximo oito caracteres. Supondo que os sensores só leem até no máximo três dígitos, então o tamanho de toda a mensagem a enviar, incluindo as vírgulas e as chavetas, será de 140bytes. Então, significa que o Synapse, incluindo o tempo de processamento dos dados e o tempo para receção do RPC para iniciar a função (usando a taxa de transferência mais baixa):

$$t_x = \frac{8 \times (12 + 140)}{250} = 4,864ms$$

$$t_{act} = 180 + 4.864 = 184.864ms$$

$$I_{act} = \frac{180 \times 20.5 + 4.864 \times 22.5}{180 + 4.864} = 20.55mA$$

$$I_{drain} = \frac{184.864}{60000} \times 20.55 + \left(1 - \frac{184.864}{60000}\right) \times 0.00037 = 0.0637mA$$

O nó principal, como tem que tratar de enviar para o Arduino toda a informação recebida de todos os nós finais, consome mais energia. O consumo de bateria depende do número de nós que o utilizador tem na sua rede.

Supondo um caso, em que o utilizador tem na sua rede de sensores, um nó principal, sem sensores acoplados, e dois nós finais com dez sensores acoplados a cada um deles. O nó principal, em primeiro lugar, tem que pedir através da função *callback* que os nós enviem os dados recolhidos. Após receber as informações é estabelecida a ligação com o Arduino, em que a porta *Serial* foi definida com uma taxa de transferência de 9600bps, mais uma vez por causa do leitor de cartões utilizado, porque à medida que a informação é recebida pelo nó principal, é enviada para o Arduino

e é guardada no cartão SD. Devido a ligação ser feita por cabo, as perdas podem ser desprezadas. Os dados que são enviados têm um tamanho de 140bytes por 10 sensores, incluindo a data e hora a que foram lidos, pois o Synapse tem um relógio de tempo real. O tamanho da *String* a enviar é agora de 161bytes por cada 10 sensores, então:

$$t_x = \frac{8 \times (12 + 322)}{250} = 10.688ms$$

$$t_{act} = 250 + 10.688 = 260.688ms$$

$$I_{act} = \frac{250 \times 20.5 + 10.688 \times 20.5}{250 + 10.688} = 20.5mA$$

$$I_{drain} = \frac{260.688}{60000} \times 20.5 + \left(1 - \frac{260.688}{60000}\right) \times 0.00037 = 0.089mA$$

Como ser observado consumo é ligeiramente maior em relação ao nó final. Mas tudo isto depende de quantos sensores vão estar inseridos nos nós, quantos nós finais existem, quantos vão fazer o encaminhamento e quanto tempo vai levar o processamento dos dados, visto que quantos mais sensores existirem mais será o tempo necessário para o Synapse enviar os dados para o Arduino.

4.4.3 Arduino UNO

O Arduino utilizado nesta dissertação também não foi o oficial. Mas tal como o Bluetooth, os consumos não diferem muito do *Datasheet* do Arduino UNO oficial.

Neste caso o Arduino vai ser dividido em dois, pois o Arduino UNO pode ser utilizado somente como programador do microcontrolador e posteriormente este ser utilizado sem auxílio do Arduino.

O Arduino funciona através de alimentação USB ou um com uma fonte externa que pode variar desde 7V a 20V, sendo que o recomendado é 12V. Possui 14 pins digitais (cujos 6 podem ser PWM) e 6 pins analógicos. O microcontrolador que está ligado ao Arduino é um ATmega328P. Tem uma memória *flash* de 32KB dos quais 0.5KB são usados para o *bootloader* e os restantes para o *script* desenvolvido pelo utilizador. A velocidade do CPU é de 16MHz. Cada vez que é ligado, este consome 50mA e por cada pino à placa é adicionado 20mA ao consumo [73]. Foi utilizado a fonte de 3.3V fornecida pela placa do Arduino.

Sendo que o Arduino tem 8 pinos ligados a ele para comunicação, com o Bluetooth, com o leitor de cartões e com o Synapse, logo o consumo instantâneo será de 210mA, e está sempre ativo, pelo

que não será uma boa escolha para um projeto onde o objetivo passa também pela durabilidade das baterias.

Assim a melhor solução é a utilização do microcontrolador em si, sendo que posteriormente é necessário inserir uma eletrónica de potência para o alimentar (figura 4.8). No caso do consumo, é bastante diferente, pois se a velocidade do microcontrolador for alterada para 1MHz, este consome 0.2mA. Uma diferença bastante significativa em relação ao Arduino.

Este terá ligada a si uma fonte de tensão, alimentada com 5V, permitindo, tal como no Arduino, a utilização dos 5V ou dos 3V para alimentação de outros dispositivos. Este continua a ter um LED de *debugging* (detetor de erros) pelo que o melhor será retirá-lo.



Figura 4.8 – Módulo para ATmega328P [74]

Supondo agora a utilização de uma bateria com capacidade igual a 2000mAh, equivalente a uma pilha AA recarregável, será então calculado uma estimativa da duração da bateria em anos.

No caso dos nós finais, usando a expressão (5):

$$Tempo = \frac{2000/(365 \times 24)}{0.0637} = 3.58 \text{ anos}$$

Como o nó principal tem o Bluetooth, o Synapse e o microcontrolador, como é de espera o seu consumo é muito maior, pelo que a duração da bateria vai ser menor:

$$Tempo = \frac{2000/(365 \times 24)}{0.158 + 0.089 + 0.2} = 0.54 \text{ anos}$$

4.5 Resultados da validação

As validações estão divididas em dois grupos, primeiro para o funcionamento de todo o sistema com três dispositivos Synapse (um nó principal e 2 nós finais), e o seguinte com a adição de sensores aos nós, verificando nos dois casos o correto envio de dados para a *Cloud*.

As validações que se seguem foram baseadas nos *Datasheets* dos fornecedores e em suposições, pois não existiu possibilidade de ter todos os equipamentos necessários, para todos os testes.

Inicialmente, vai ser demonstrado como é adicionado um utilizador e um sensor com a aplicação criada e, posteriormente, o conceito e todo o percurso desde o sensor até ao telemóvel.

4.5.1 Montagem do protótipo

Os equipamentos que foram necessários para provar o conceito desta dissertação foram:

- Arduino;
- Telemóvel (Smartphone) Android com Bluetooth;
- Leitor de cartões SD;
- Cartão SD (neste caso de 4GB);
- Bluetooth (Modelo HM-10);
- LED.

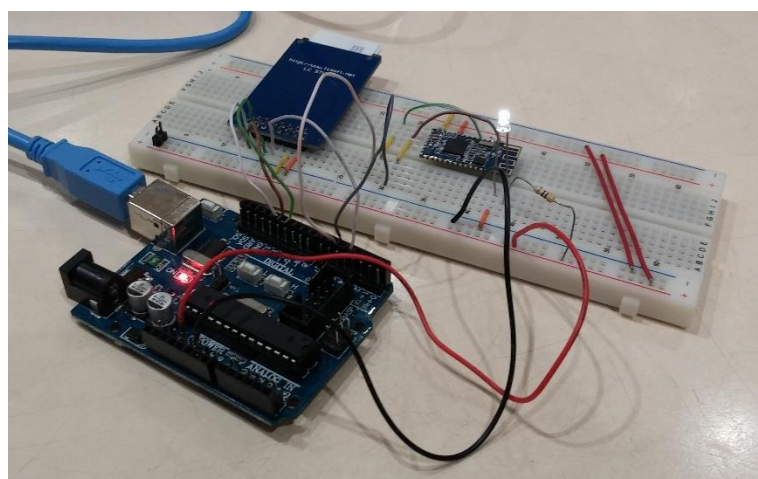


Figura 4.9 - Protótipo

O Bluetooth é ligado aos pinos 5 e 6 que, através da biblioteca *Serial*, esses pinos vão ser transformados em portas *Seria*. O leitor de cartões SD tem por pré definição da biblioteca SD e SPI os pinos 11, 12 e 13 que corresponde respetivamente a MOSI, MISO e SCK. E o pino CS foi ligado por opção ao pino 4. O LED (servirá apenas para deteção de erros) está ligado ao módulo Bluetooth que vai indicar quando é que ele está desligado, à procura de dispositivos e quando está emparelhado, como pode ser verificado na figura 4.9.

4.5.2 Comunicação com o Serviço Web

A única comunicação existente com o Serviço Web é quando o utilizador faz o seu registo, ou quando quer registar um novo sensor.

Ao fazer um novo registo de utilizador é necessário primeiro carregar no botão “*Regist*” e, posteriormente, inserir o nome de utilizador, uma palavra passe e carregar no botão “*Send*”, aguardando a mensagem a indicar se o registo foi feito com sucesso ou não (figura 4.10).

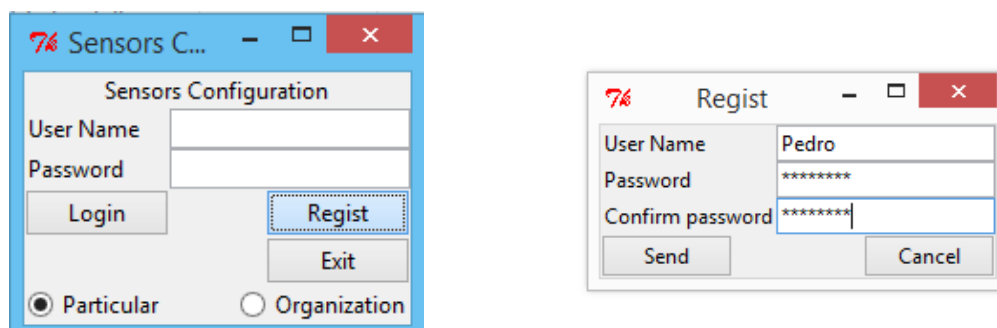


Figura 4.10 - Exemplo de um registo de um novo utilizador

A forma mais fácil de se observar que o registo foi feito, é através da página de internet criada apresentado na figura 4.11.

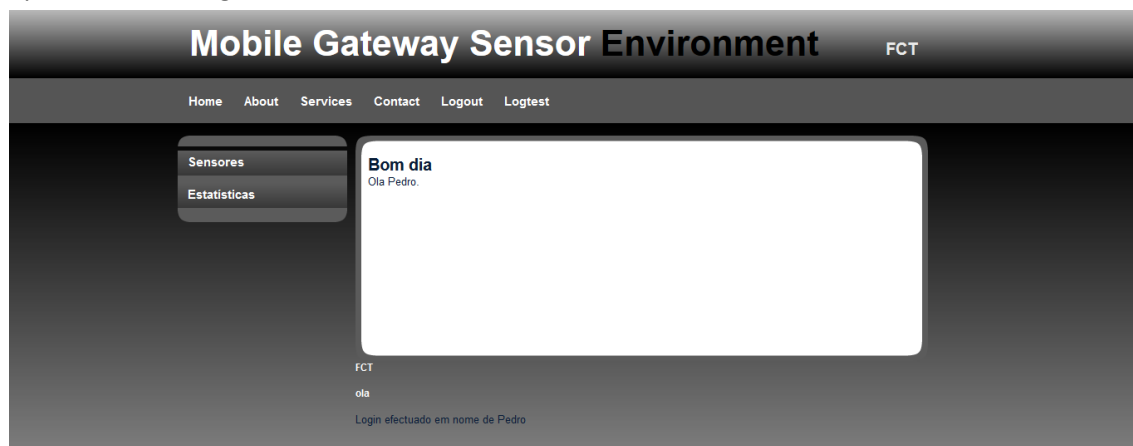


Figura 4.11 - Web Site criado na dissertação Ponto de Acesso Móvel em Ambiente Sensorial [75]

Quando o utilizador estiver registado, poderá ver os seus sensores e, caso pretenda, adicionar um novo sensor. A diferença do Login de um utilizador particular para um utilizador de uma organização é apresentada na figura 4.12.

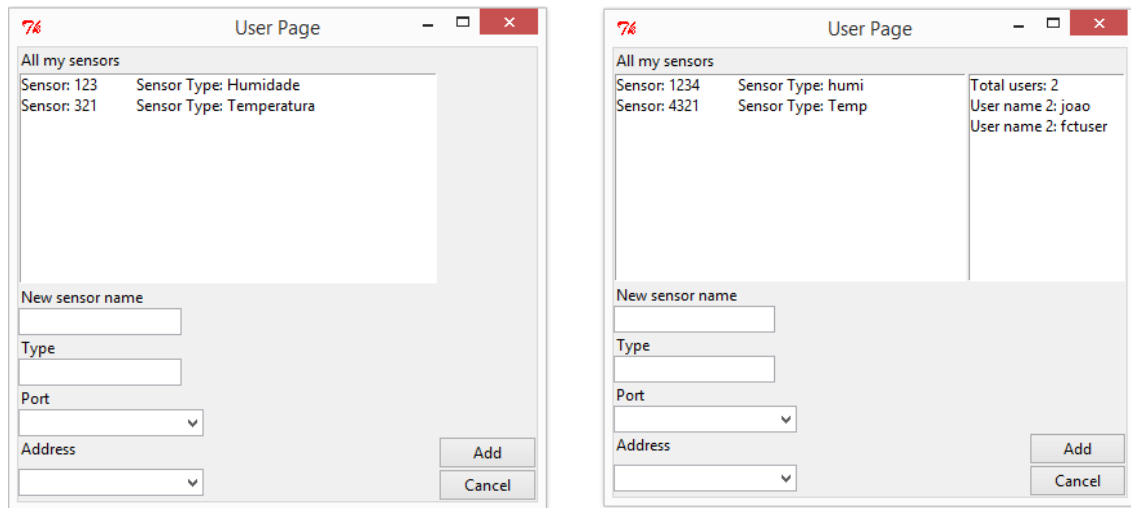


Figura 4.12 - Diferenças de um utilizador particular (lado esquerdo) e de uma organização (do lado direito)

Como se pode verificar na figura 4.12, do lado direito, é uma organização, onde pode ter acesso a quem são os utilizadores que pertencem à mesma, sendo, neste caso, a organização é a FCT e os utilizadores que lhe pertencem são o joao e o fctuser.

Supondo agora, que o utilizador particular quer adicionar um novo sensor com o nome TempCasa e o tipo é Temperatura, então preenche os campos necessários e carrega no botão “Add” e caso tenha o nó principal ligado ao computador, o sensor é adicionado, como pode ser verificado na figura 4.13.

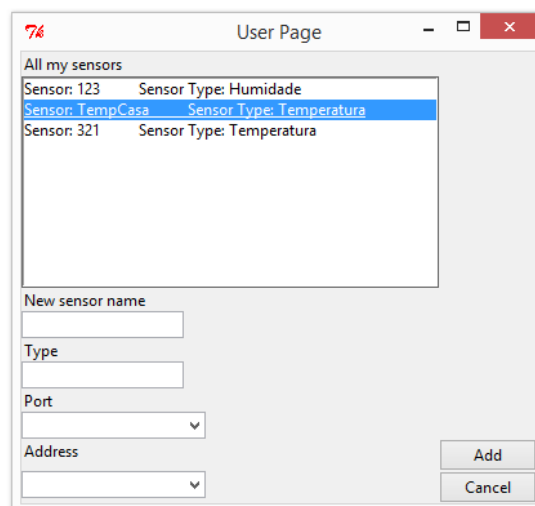


Figura 4.13 - Exemplo da adição de um novo sensor

Com o utilizador e o sensor TempCasa registados será feito o primeiro teste de validação do conceito.

4.5.3 Teste de operação

Comunicação entre Synapse e Arduino

Não foi possível demonstrar como funcionaria a rede em Malha com os dispositivos Synapse pois não foi possível ter os equipamentos. Mas supondo que o utilizador já tem o seu sensor TempCasa instalado no nó final da rede. A cada minuto, o sensor recolhe informação e o nó final envia essa informação para o nó principal, que comunica com o Arduino e guarda essa informação no cartão SD. A figura 4.14 representa como é guardada a informação.

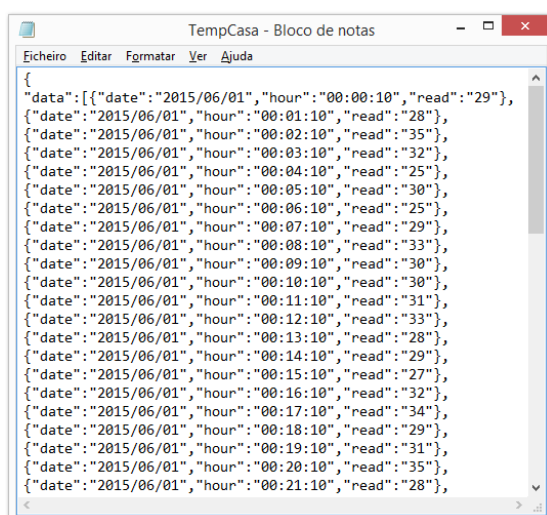


Figura 4.14 - Ficheiro .txt que representa os valores recebidos da rede de sensores

O ficheiro acima foi criado com dados aleatórios e aumentando um minuto a cada leitura. Este tem que cumprir rigorosamente esta sequência e, como é possível observar, a data e a hora tem também que seguir essa formatação ou seja YYYY/MM/DD e HH/MM/SS.

Comunicação entre Arduino e Telemóvel

Quando é estabelecida uma comunicação entre o Arduino e um telemóvel, todas as outras tentativas de ligação ao Arduino vão ser rejeitadas. No caso de teste, o utilizador tem que ligar o Bluetooth e conectar-se ao nó principal, pois o Bluetooth utilizado não possui a possibilidade de funcionar como mestre e procurar todos os dispositivos Bluetooth à sua volta.

Neste tópico, o telemóvel e o Arduino serão submetidos a três testes. O primeiro será um simples envio de dados de tamanho curto, o segundo será quando se tratará de um envio de uma hora

completa de um sensor, ou seja, o ficheiro com sessenta minutos de informação já organizados e, o terceiro servirá para apagar do cartão SD a data mais recente que esteja na base de dados na *Cloud*.

O primeiro teste passa simplesmente por mostrar a comunicação entre os Bluetooth e o Arduino. Isto pode acontecer de duas formas, uma quando a rede de sensores foi instalada recentemente (ou seja, não existem muitos dados a enviar para o utilizador) ou quando a hora em que o ficheiro foi criado ainda não estiver no fim.

Foi usada uma aplicação em Android para testar a comunicação, mas nas descrições seguintes, é assumido que a aplicação funciona automaticamente, sem ser necessário a intervenção do utilizador. Essa aplicação (figura 4.15) tem um botão “*Connect*”, que serve para enviar um código que demonstra que o utilizador tem a aplicação instalada e um botão “*Read*” que envia um código para informar o Arduino que está pronto para a receção de dados do ficheiro mais antigo.

Assim que o utilizador liga o Bluetooth para estabelecer a comunicação este envia o código de autenticação e após ter sido confirmado pelo Arduino é enviado o código que indica que irá receber a data mais recente do sensor que está na base de dados da *Cloud*.

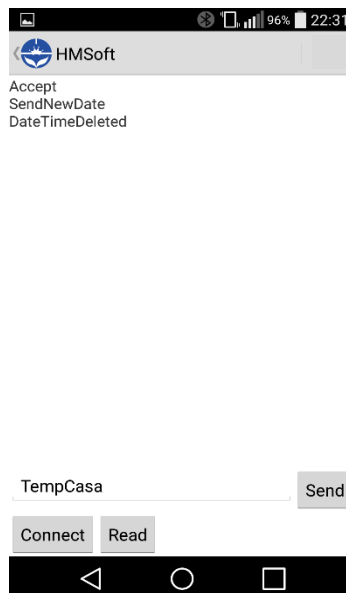


Figura 4.15 - Exemplo da atualização do cartão SD (envio de dados na aplicação Android)

Na figura 4.15, pode-se observar a sequência acima indicada, em que as mensagens escritas indicam a confirmação por parte do Arduino. O *Accept* é a resposta à palavra passe enviada para aceder aos dados dos sensores, o *SendNewDate* é a resposta ao código enviado sobre a data mais recente na base de dados e o *DateTimeDeleted* é a resposta final do Arduino que avisa a aplicação

de que foi eliminado com sucesso os dados mais antigos do sensor específico (TempCasa) do cartão SD. Com este processo fica demonstrando o terceiro teste.

De seguida, é enviado um novo código ao Arduino que pede que envie o ficheiro mais antigo presente no cartão SD.

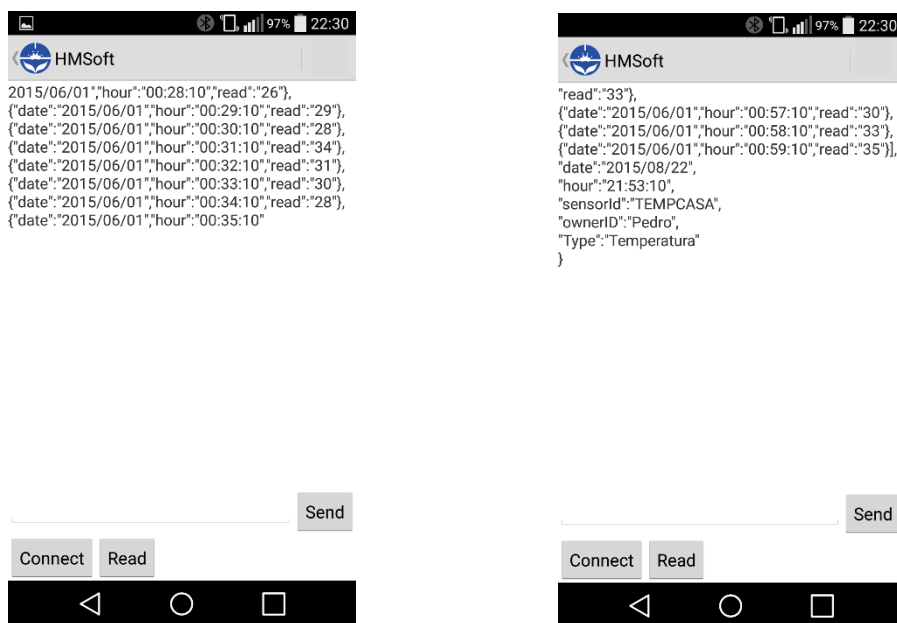


Figura 4.16 - Exemplo da receção de dados enviados pelo Bluetooth

Pode ser comparada a informação recebida na aplicação, presente na figura 4.16, com a Figura 4.14, que indica os dados recebidos na aplicação e os enviados pelo Arduino, respetivamente. A imagem tem dados repartidos pois a aplicação não foi desenhada com a função *scroll*, pelo que quando chega ao fim do ecrã apaga tudo e volta a escrever.

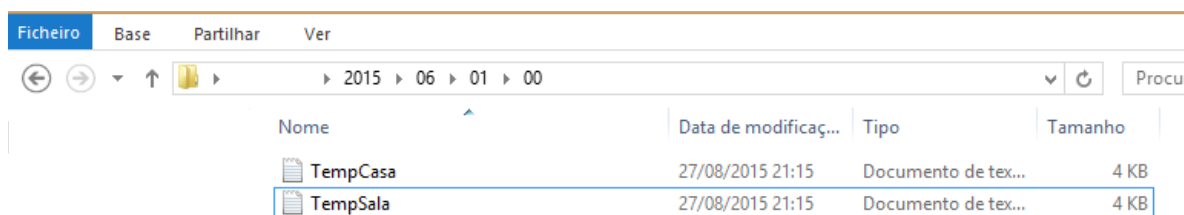
Como pode ser verificado, a receção dos dados é correta e recebida em formato JSON, em que no fim apresenta a data e a hora a que foi enviado, o nome do sensor, o utilizador e o tipo de sensor que é.

Quando o ficheiro é recebido na totalidade, é finalizada a conexão entre o Bluetooth e o Arduino. Provando assim, que o conceito pretendido funciona.

4.5.4 Teste de expansão

No segundo teste, mais uma vez, não foi possível testar a rede em Malha para se perceber como é que iria reagir a dois sensores ligados, com o envio de dados simultaneamente. Em relação ao Arduino e ao telemóvel a forma de testar, passa pela criação de mais um ficheiro com dados

aleatórios, que representa a adição de um novo sensor, neste caso denominou-se TempSala. Todos os testes são executados da mesma forma com a exceção de quando é pedido para eliminar os dados mais antigos do sensor TempCasa, o ficheiro TempSala passa a ser o ficheiro mais antigo. Isto acontece caso o TempCasa e o TempSala iniciem a sua leitura de dados praticamente ao mesmo tempo (na mesma hora), criando assim, dois ficheiros. Estes ficheiros, por consequência são adicionados na mesma diretoria como mostra a figura 4.17.



Nome	Data de modificaç...	Tipo	Tamanho
TempCasa	27/08/2015 21:15	Documento de tex...	4 KB
TempSala	27/08/2015 21:15	Documento de tex...	4 KB

Figura 4.17 – Diretorias criadas no cartão SD com formato YYYY/MM/DD/HH

5

Conclusões e Trabalhos Futuros

5.1 Conclusões

O projeto tinha como objetivo o desenvolvimento de um produto/conceito que procedesse à recolha de informação através de sensores que pudessem ser instalados em qualquer lugar, sem que estivessem dependentes de pontos fixos com internet, de operadoras móveis para enviar os seus dados através delas e de uma instalação elétrica, para posteriormente poderem ser acedidos pelo utilizador através da *Cloud*.

Todos os objetivos acima referidos foram cumpridos e foi comprovado o conceito, pois o utilizador pode instalar os seus sensores num local pretendido, ficando apenas dependente do número de pessoas que possuem um telemóvel (*smartphone*) com Bluetooth e ligação à internet, do número de pessoas que passem constantemente pelo nó principal de forma a recolherem a sua informação dos sensores e fica dependente do tempo que a bateria acoplada ao sensor leva para se esgotar.

De salientar que o grande problema da proposta desenvolvida nesta dissertação está relacionado com o caso de nenhuma pessoa se aproximar do nó principal de forma a recolher os dados que, por sua vez, vai fazer com que o cartão fique completamente cheio. Por consequência, a recolha de informação dos sensores pára a partir desse momento. No caso de um utilizador desatendo, não irá perceber que o sensor deixou de enviar dados para a *Cloud*.

De acordo com os resultados obtidos em fase de experimentação, é verificado que a rede em Malha não pôde ser testada. Seria de esperar que tudo funcionasse corretamente, caso o dispositivo escolhido Synapse esteja de acordo com o que é descrito no seu *Datasheet* e no *white-paper*.

Por também terem sido utilizados dispositivos não oficiais (ou seja, foram utilizadas imitações, pois eram monetariamente mais acessíveis), estes não funcionaram como seria de esperar, como é o caso do Bluetooth e do Arduino.

O Bluetooth utilizado só funciona como mestre caso se queira encontrar dispositivos do mesmo modelo, o que implica, como foi explicado anteriormente, o utilizador ao invés de aguardar que o Bluetooth lhe envie uma mensagem para emparelhamento, terá de ser o próprio a fazê-lo.

No caso do Arduino, ao fim de muitas tentativas e *scripts* enviados, a memória interna ficou corrompida, pelo que se não se exceder um certo tamanho de memória tudo funciona corretamente. Quando é feito o *upload* do ficheiro *script* completo, certas funções deixam de funcionar.

De acordo com os dados recolhidos pelos *Datasheet* dos fornecedores, verificou-se que os nós finais, com períodos de funcionamento de um minuto e com a utilização de uma pilha AA (2000mAh), são capazes de funcionar aproximadamente três anos e seis meses, cada um. Já o nó principal, como seria de esperar, por ter agregado a si mais dispositivos que os nós finais, com a mesma pilha AA, dura apenas seis meses. Pelo que seria melhor, em vez da utilização de uma só pilha, acrescentar, por exemplo mais uma pilha AA, o que aumentaria para o dobro, ou seja, um ano. Outra hipótese seria baixar a potência da antena do Bluetooth de modo que o consumo seja mais baixo, que em contrapartida o alcance será reduzido.

Foi tido em conta a utilização de uma rede em Malha para assegurar que se um dos nós deixar de funcionar, o utilizador poderá receber a informação dos outros nós através do algoritmo dos mesmos. Mas, por outro lado, não foi tido em conta como se poderá resolver o projeto, caso o nó principal deixe de funcionar.

A respeito de ser um projeto feito em parceria com empresa IrRADIARE, foi também cumprido o pretendido. Pois como a IrRADIARE lida com problemas ambientais, pode assim instalar os sensores (após as atualizações referidas nos trabalhos futuros) e fornecer (a custo ou não) os dados a principais interessados, de modo a que estes tomem medidas para melhorar o ambiente.

5.2 Trabalhos Futuros

Para uma melhor consistência do funcionamento do sistema, serão necessárias algumas intervenções no projeto.

Assumindo que o utilizador recebe a data (mais recente na *Cloud*) de um sensor que pertence a uma área A, através do seu telemóvel, mas que o mesmo resida na área B, muito provavelmente não vai conseguir atualizar o cartão SD do nó principal onde está esse sensor. Assim a melhor forma seria, quando um utilizador se aproximasse de um nó principal, o Arduino possuiria um ficheiro com toda a informação dos nós, o nome dos sensores e o nome do utilizador daquela rede. Possibilitando que, quando a comunicação fosse estabelecida, o Arduino enviaria esse ficheiro para o utilizador. Quando o utilizador ligasse o telemóvel à internet, iria ser devolvido a data mais recente dos sensores incluídos no ficheiro recebido anteriormente, evitando que tenha informações de sensores que, muito provavelmente, nunca iria conseguir atualizar a informação do nó principal onde estão inseridos.

Outra alteração a ser feita seria na parte de leitura dos sensores, pois todos os sensores ligados a um nó final recolhem a sua informação de minuto a minuto. Isto não seria necessário, por exemplo, para a leitura da temperatura de uma certa localidade, mas a leitura da temperatura de um microcontrolador já poderia ser mais sistemática. O modo de contornar essa situação seria, quando o utilizador registasse os seus sensores através da aplicação, criar um campo, onde se possa inserir também a frequência com que deve ser feita a leitura, que serão guardados nos parâmetros de memória do Synapse. Como este possui um relógio em tempo real, que não pára de funcionar em modo *Sleep*, após passar o tempo necessário para um determinado sensor, o Synapse entra em modo Ativo, lê e envia a informação para o nó principal.

Seria também importante, quando a memória do cartão ficar cheia, existir uma função que possa verificar se não existirá informações mais importantes do que as que estão a ocupar o espaço no cartão SD. Esta função poderá ser de prevenção, impedindo que informações desnecessárias ocupem espaço. No caso de um sensor de temperatura se, por exemplo, esta se mantiver inalterada durante um dia completo, não seria necessário guardar minuto a minuto essa informação, mas sim o início e o fim dessa temperatura. Se a função for de aviso, como por exemplo, dar importância quando o monóxido de carbono atingiu valores perigosos, esta substitui um ficheiro desnecessário, de modo a que o utilizador possa ter conhecimento dessa ocorrência.

O acesso ao cartão SD foi limitado aos 9600bps e, para que o Bluetooth envie os dados mais rapidamente para o telemóvel, será melhor escolher um leitor de cartões que tenha uma

velocidade de leitura significativamente maior. Por conseguinte, aumentando a taxa de transferência, o Bluetooth ficará muito menos tempo ativo, prolongando o tempo de vida das baterias. [75]

Bibliografia

- [1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam e E. Cayirci, "Wireless sensor networks: a survey," pp. 1-9, 20 Dezembro 2001.
- [2] M. Yuriyama e T. Kushida, "Sensor-Cloud Infrastructure-Physical Sensor Management with Virtualized Sensors on Cloud Computing," em *13th International Conference on Network-Based Information Systems*, Japão, 2010.
- [3] M. Rouse, "WhatIs," Junho 2014. [Online]. Available: <http://whatis.techtarget.com/definition/network-topology>. [Acedido em 23 Julho 2015].
- [4] L. Antunes, "Identificação de pessoas numa portaria virtual," DEETC, ISEL, Lisboa, PT, 2012.
- [5] P. Pinto, "pplware no comments," 23 Dezembro 2010. [Online]. Available: <http://pplware.sapo.pt/tutoriais/networking/lan-man-wan-pan-san-%E2%80%A6-sabe-a-diferenca/>. [Acedido em 20 Fevereiro 2015].
- [6] R. Russo, "escreveassim," 17 Abril 2012. [Online]. Available: <http://escreveassim.com.br/2012/04/17/redes-lan-man-wan-pan-san-can-wman-wwan-e-ran-qual-a-diferenca/>. [Acedido em 20 Fevereiro 2015].
- [7] "Technopedia," Janalta Interactive Inc., [Online]. Available: <https://www.techopedia.com/definition/5107/wireless-local-area-network-wlan>. [Acedido em 28 Agosto 2015].
- [8] S. Klaysov, "ITworld," 20 Fevereiro 2013. [Online]. Available: <http://itworldguide.com/types-of-wireless-networks/>. [Acedido em 20 Fevereiro 2015].
- [9] "Bluetooth," Bluetooth SIG, [Online]. Available: <http://www.bluetooth.com/Pages/low-energy-tech-info.aspx>. [Acedido em 3 Fevereiro 2015].
- [10] P. Zandbergen, "Study," [Online]. Available: <http://study.com/academy/lesson/short-range-wireless-communication-bluetooth-zigbee-infrared-transmission.html>. [Acedido em 23 Junho 2015].
- [11] E. Alecrim, "Info Wester," 24 Junho 2013. [Online]. Available: <http://www.infowester.com/wifi.php>. [Acedido em 28 Julho 2015].
- [12] A. Ganhão, "On the Cloud Deployment of a Session Abstraction for Service/Data Aggregation," DEE, FCT-UNL, Monte de Caparica, PT, 2010.
- [13] L. Frenzel, "electronic design," Penton, 11 Outubro 2012. [Online]. Available: <http://electronicdesign.com/communications/fundamentals-short-range-wireless-technology>. [Acedido em 16 Setembro 2015].

- [14] A. Buczkowski, "Masters Program in Geospatial Technologies," FCT-UNL, Monte de Caparica, PT, 2012.
- [15] "gsmarena," [Online]. Available: <http://www.gsmarena.com/glossary.php3?term=irda>. [Acedido em 23 Junho 2015].
- [16] R. Laires, "Sistema de iluminação eficiente utilizando a tecnologia LED para espaços públicos interiores," DEE, FCT-UNL, Monte de Caparica, PT, 2013.
- [17] "Digi," [Online]. Available: http://www.digi.com/pdf/wp_zigbee.pdf. [Acedido em 14 Março 2015].
- [18] G. Patrício, "Rede Sem Fios de Microcontroladores com Acesso Remoto Aplicada à Domótica," DEE, FCT-UNL, Monte de Caparica, PT, 2009.
- [19] D. Ewing, "Synapse," 2015. [Online]. Available: <http://www.synapse-wireless.com/upl/downloads/industry-solutions/reference/white-paper-synapse-snap-scalability-b35afb20.pdf>. [Acedido em 29 Julho 2015].
- [20] D. Prindle, "Digital Trends," 31 Janeiro 2014. [Online]. Available: <http://www.digitaltrends.com/home/zigbee-vs-zwave-vs-insteon-home-automation-protocols-explained/>. [Acedido em 29 Julho 2015].
- [21] C. S. Raghavendra, K. M. Sivalingam e T. Znati, em *Wireless Sensor Networks*, EUA, Electronic Services, 2004, pp. 8-12; 60-66.
- [22] A. S. Q. Rito, *Redes de Sensores Sem Fios*, Lisboa, 2006.
- [23] V. Lins e W. Moura, "DOMÓTICA: AUTOMAÇÃO RESIDENCIAL," p. 1.
- [24] J. Almeida, "Intrusion Tolerant Routing with Data Consensus in Wireless Sensor," DI, FCT-UNL, Monte de Caparica, PT, 2013.
- [25] cshanika, "Winstudent," 14 Junho 2015. [Online]. Available: <http://www.winstudent.com/mps-net/>. [Acedido em 20 Agosto 2015].
- [26] N. Boavida, "Pesquisa Segura de Dados em Redes de Sensores," DI, FCT-UNL, Monte de Caparica, PT, 2010.
- [27] T. Agarwal, "ELPROCUS," [Online]. Available: <https://www.elprocus.com/what-is-zigbee-technology-architecture-and-its-applications/>. [Acedido em 29 Julho 2015].
- [28] A. Thakur, "Engineers Garage," [Online]. Available: <http://www.engineersgarage.com/articles/what-is-zigbee-technology?page=3>. [Acedido em 29 Julho 2015].
- [29] "Jennic," 2007. [Online]. Available: <http://www.jennic.com/elearning/zigbee/files/html/module3/module3-2.htm>. [Acedido em 29 Julho 2015].

- [30] S. K. Singh, M. P. Singh e D. K. Singh, "Routing Protocols in Wireless Sensor Networks - A Survey," *International Journal of Computer Science & Engineering Survey (IJCES)*, vol. 1, nº 2, pp. 67-70, 2010.
- [31] W. Ahmad e M. K. Aslam, "An Investigation of Routing Protocols in Wireless Mesh Network under certain Parameters," BTH, Suécia.
- [32] J. N. Al-Karaki e A. E. Kamal, *ROUTING TECHNIQUES IN WIRELESS SENSOR NETWORKS: A SURVEY*, EE Department-Stanford University, 2004.
- [33] "Telematics Computer Systems," [Online]. Available: <http://www.des-testbed.net/content/ad-hoc-demand-distance-vector-routing-aadv>. [Acedido em 7 Setembro 2015].
- [34] "The Internet Society," 2005. [Online]. Available: <http://moment.cs.ucsb.edu/pub/draft-ietf-manet-dymo-00.html#toc>. [Acedido em 4 Setembro 2015].
- [35] M. Rouse, "TechTarget," Outubro 2005. [Online]. Available: <http://searchnetworking.techtarget.com/definition/Link-Quality-Source-Routing>. [Acedido em 4 Setembro 2015].
- [36] Z. Hong-tu e M. Yue-qi, "Improved Routing Algorithm Research for ZigBee Network," *Proceedings of the Third International Symposium on Computer Science and Computational Technology*, pp. 17-20, 14-15 Agosto 2010.
- [37] K. Machado, D. Rosário, E. Cerqueira, A. A. F. Loureiro, A. Neto e J. N. d. Souza, "A Routing Protocol Based on Energy and Link Quality for Internet of Things Applications," *sensors*, vol. 13, p. 1948, 2013.
- [38] "How-To Geek," [Online]. Available: <http://www.howtogeek.com/167783/htg-explains-the-difference-between-wep-wpa-and-wpa2-wireless-encryption-and-why-it-matters>. [Acedido em 30 Julho 2015].
- [39] A. Pearson, "Security Innovation Europe," 18 Dezembro 2014. [Online]. Available: <http://www.securityinnovationeurope.com/blog/whats-the-difference-between-hashing-and-encrypting>. [Acedido em 27 Agosto 2015].
- [40] "TechTarget," Novembro 2014. [Online]. Available: <http://searchsecurity.techtarget.com/definition/Advanced-Encryption-Standard>. [Acedido em 27 Agosto 2015].
- [41] "Digicert," [Online]. Available: <https://www.digicert.com/ssl.htm>. [Acedido em 14 Agosto 2015].
- [42] S. Basagni, M. Y. Naderi, C. Petrioli e D. Spenza, "WIRELESS SENSOR NETWORKS WITH ENERGY HARVESTING," pp. 2-5.
- [43] R. Allan, "ENERGY HARVESTING POWERS INDUSTRIAL WIRELESS SENSOR NETWORKS," *EngineeringFeature*, pp. 23-29, 20 Setembro 2012.

- [44] C.-Y. Chong e S. P. Kumar, "Sensor Networks: Evolution, Opportunities, and Challenges," *IEEE*, vol. 91, nº 8, pp. 1251-1254, Agosto 2003.
- [45] "The Evolution of Wireless Sensor Networks," *Silicon Labs*, 2013.
- [46] "Engineers Garage," [Online]. Available: <http://www.engineersgarage.com/articles/sensors?page=1>. [Acedido em 13 Junho 2015].
- [47] "Sparkfun," Sparkfun Electronics, [Online]. Available: <https://www.sparkfun.com/products/251>. [Acedido em 16 Junho 2015].
- [48] "TSMC Thermo Supply," [Online]. Available: <http://www.thermosupply.com.br/ecom.aspx/Produto/termopar-mineral,-tipo-k,-pote-e-rabicho-6x300mm>. [Acedido em 22 Agosto 2015].
- [49] "Direct Industry," [Online]. Available: <http://www.directindustry.com/prod/conax-technologies/product-7421-169063.html>. [Acedido em 20 Agosto 2015].
- [50] "Electrodex," [Online]. Available: <http://www.eletródex.com.br/termistor-ntc-5mm.html>. [Acedido em 20 Agosto 2015].
- [51] "LC Resources," LC Resources Inc., [Online]. Available: <http://www.lcresources.com/resources/getstart/2e01.htm>. [Acedido em 13 Junho 2015].
- [52] "T UVHaisen," Haisen Technik Co., Ltd, [Online]. Available: <http://www.haisen.org/download/>. [Acedido em 17 Setembro 2015].
- [53] "Safewife," [Online]. Available: <http://www.safewise.com/home-security-faq/carbon-monoxide-detector>. [Acedido em 4 Setembro 2015].
- [54] "Multilógica," [Online]. Available: <https://multilogica-shop.com/Sensor-de-g%C3%A1s-mon%C3%B3xido-de-carbono-MQ-7>. [Acedido em 20 Agosto 2015].
- [55] J. Domingos, "On the Cloud Deployment of a Session Abstraction for Service/Data Aggregation," DI, FCT-UNL, Monte de Caparica, PT, 2013.
- [56] "IDC," IDC Corporate USA, [Online]. Available: <http://www.idc.com/prodserv/smartphone-os-market-share.jsp>. [Acedido em 6 Junho 2015].
- [57] S. Kovach, "Business Insider," 13 Agosto 2013. [Online]. Available: <http://www.businessinsider.com/history-of-android-2013-8?op=1>. [Acedido em 15 Julho 2015].
- [58] "Android," [Online]. Available: <http://www.android.com/history/>. [Acedido em 15 Julho 2015].
- [59] "amazon web services," 24 Agosto 2006. [Online]. Available: <https://aws.amazon.com/pt/about-aws/whats-new/2006/08/24/announcing-amazon-elastic-compute-cloud-amazon-ec2---beta/>. [Acedido em 23 Agosto 2015].

- [60] Tomtsongas, "Program Success," 27 Fevereiro 2015. [Online]. Available: <https://programsucces.wordpress.com/2015/02/27/management-of-cloud-projects/>. [Acedido em 30 Abril 2015].
- [61] "Valarm," [Online]. Available: <http://www.valarm.net/>. [Acedido em 5 Setembro 2015].
- [62] "SensorCloud," LORD MicroStrain, [Online]. Available: <http://www.sensorcloud.com/documentation>. [Acedido em 5 Setembro 2015].
- [63] "Libelium," Libelium Comunicaciones Distribuidas S.L, [Online]. Available: <http://www.libelium.com/products/waspmote-mote-runner-6lowpan/>. [Acedido em 5 Setembro 2015].
- [64] A. Oliveira, *Rdes Móveis-Aula 3*, DEE FCT/UNL.
- [65] "Synapse-Wireless," [Online]. Available: <http://www.synapse-wireless.com/upl/downloads/industry-solutions/reference/reference-manual-snap-os-45352fdc.pdf>. [Acedido em 29 Julho 2015].
- [66] "Synapse-Wireless," 2015. [Online]. Available: <http://www.synapse-wireless.com/upl/downloads/industry-solutions/reference/product-brief-snap-os-b89f38cf.pdf>. [Acedido em 29 Julho 2015].
- [67] "Solarbotics," [Online]. Available: <https://solarbotics.com/product/39252/>. [Acedido em 4 Setembro 2015].
- [68] [Online]. Available: <http://www.seeedstudio.com/depot/Grove-Bee-Socket-p-1449.html>. [Acedido em 4 Setembro 2015].
- [69] "PTROBOTICS," [Online]. Available: http://www.ptrobotics.com/shields-comunicacao/1087-arduino-wireless-sd-shield.html?search_query=PTR001087&results=1. [Acedido em 4 Setembro 2015].
- [70] E. Casilari, J. M. Cano-Garcia e G. Campos-Garrido, "Modeling of Current Consumption in 802.15.4/ZigBee Sensor Motes," *MDPI*, pp. 5457-5459, 1 Junho 2010.
- [71] "TinySine," Tinyos, [Online]. Available: www.tinyosshop.com/index.php?route=product/product&product_id=705. [Acedido em 2 Setembro 2015].
- [72] "Synapse-Wireless," [Online]. Available: <http://www.synapse-wireless.com/upl/downloads/industry-solutions/reference/datasheet-synapse-rf-engine-rf200p81-rf200pu1-c6bd5719.pdf>. [Acedido em 7 Setembro 2015].
- [73] "Arduino," [Online]. Available: <https://www.arduino.cc/en/Main/arduinoBoardUno>. [Acedido em 7 Setembro 2015].
- [74] "adafruit," [Online]. Available: <https://www.adafruit.com/products/91>. [Acedido em 10 Setembro 2015].

[75] A. José, “Ponto de Acesso Móvel em Ambiente Sensorial,” DEE, FCT-UNL, Monte de Caparica, PT, 2015.

Anexo

Synapse SNAP Modules	Through-Hole Modules						Surface-Mount	
	RF200P81	RF200PF1	RF200PD1	RF220UF1	RF266PC1	RF266PU1	SM220UF1	SM200P81
User Application Scripts Uploadable Over-the-Air	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Core SNAP Network Upgradeable Over-the-Air	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Frequency Band	2.4 GHz	2.4 GHz	2.4 GHz	2.4 GHz	2.4 GHz	2.4 GHz	2.4 GHz	2.4 GHz
Raw Bandwidth	250Kbps, 500Kbps, 1Mbps, 2Mbps	250Kbps, 500Kbps, 1Mbps, 2Mbps	250Kbps, 500Kbps, 1Mbps, 2Mbps	250Kbps, 500Kbps, 1Mbps, 2Mbps	1Mbps, 2Mbps	250Kbps, 500Kbps, 1Mbps, 2Mbps	250Kbps, 500Kbps, 1Mbps, 2Mbps	1Mbps, 2Mbps
Memory Size	128K	128K	128K	128K	128K	128K	128K	128K
RAM Size	16K	16K	16K	16K	16K	16K	16K	16K
Antenna	Chip	F Antenna	RP-SMA	U.FL & Compact F	Chip	U.FL	U.FL & Compact F	Chip
Receive Amp	No	Yes (Discrete)	Yes (Discrete)	Yes (FEM)	Yes (Discrete)	Yes (Discrete)	Yes (FEM)	No
Transmit Amp	No	Yes (Discrete)	Yes (Discrete)	Yes (FEM)	Yes (Discrete)	Yes (Discrete)	Yes (FEM)	No
Transmit Power Output	+3 dBm	+15 dBm	+15 dBm	+20 dBm	+15 dBm	+15 dBm	+20 dBm	+3 dBm
Receiver Sensitivity	-100 dBm	-103 dBm	-103 dBm	-103 dBm	-100 dBm	-100 dBm	-103 dBm	-100 dBm
Processor Size	8 bit	8 bit	8 bit	8 bit	8 bit	8 bit	8 bit	8 bit
Size	33.86mm x 33.86mm	33.86mm x 33.86mm	33.86mm x 33.86mm	33.86mm x 33.86mm	24.38mm x 32.94mm	24.38mm x 32.94mm	29.8mm x 19mm	29.8mm x 19mm
Distance (Open Field)	1500ft	TBD	TBD	3 mi/2500ft	4000ft	1 mi	3 mi/2500ft	1500ft
Temperature Range	-40°C / +85°C	-40°C / +85°C	-40°C / +85°C	-40°C / +85°C	-40°C / +85°C	-40°C / +85°C	-40°C / +85°C	-40°C / +85°C
Certifications	FCC, IC	FCC, IC	FCC, IC	FCC, IC	FCC, IC	FCC, IC	FCC, IC, CE	FCC, IC
I/O Pins	20	20	20	20	15	15	33	33
A/D Pins	8	8	8	8	4	4	8	8
A/D Bits	10	10	10	10	10	10	10	10
Memory available for applications	58.5K	58.5K	58.5K	58.5K	58K	58K	58.5K	58.5K
Basic encryption	Included	Included	Included	Included	Included	Included	Included	Included
AES encryption	Included	Included	Included	Included	Included	Included	Included	Included

Figura A.1 – Tabela de comparação de dispositivos Synapse